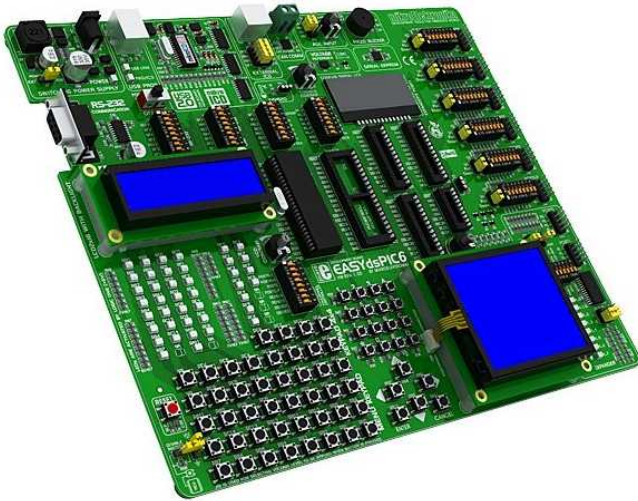
型番:dsPICF-400　　　　REV.1.00

USB2.0対応　高速プログラマ搭載
dsPIC30Fシリーズ開発用統合評価ボード

4-C Version

取扱説明書
お使いになる前にこの説明書をよくお読みの上正しくお使いください。

ボード本体

※上記写真は、オプションのGLCDを装着した例です。

製品の概要

dsPIC30Fシリーズ開発用統合評価ボード(型番:dsPICF-400、以下dsPICF-400と記載)は、オンボードでUSB2.0対応のプログラマーを搭載し、最新のDSP機能搭載16ビットマイコン、dsPIC30シリーズに対応した統合開発ボードです。

ボード上には、18ピン、28ピン、40ピンのソケットがあり、ピン数の異なるdsPICでもそのまま搭載して開発することができます。また、ピンの配置が異なる28ピンのデバイスも装着できるよう、3つのソケットを用意しました。

ボード上に搭載のUSB接続マイコンライターは、現在流通しているほぼすべてのdsPIC30Fシリーズに対応しており、基板上に装着したマイコンにプログラムを書き込みます。USBバスからdsPICF-400の電源を取りますので、別途電源を準備する必要がありません。

ボードに搭載されている各周辺回路に接続されているピンは周辺回路に接続して利用するか又は周辺回路には接続しないで使用するかをディップスイッチやジャンパーソケットで選択できます。また全I/Oピンはボード右側に実装されているヘッダピンから取り出すことができます。また、本セットには6Kワードまでのhexファイルが生成できるICD機能付きの体験版Cコンパイラーを付属。本書にはチュートリアルも収録されているので、すぐに16ビットマイコンの世界を体験できます。

パッケージの内容

■同梱物
- dsPICF-400ボード本体
- dsPIC30F4013-30I/P　（ボードに装着済み）
- 16文字×2行液晶ディスプレイモジュール　（ボードに装着済み）
- USBケーブル
- CD-ROM
- マニュアル(本書)

ドライバのインストール及びパソコンとの接続

■デバイスドライバーのインストール

dsPICF-400をパソコンと接続する前に、デバイスドライバーをパソコンにインストールします。
（Windows98及びMEをご使用のお客様は本項は読み飛ばして、次の「パソコンと接続する」の項よりお読み下さい。）
WindowsXP.Vista,7をご利用のお客様は下記の手順で、ドライバーをインストールします。

1 付属のCD-ROMをパソコンのCD-ROMドライブに挿入してエクスプローラー等で内容を表示します。

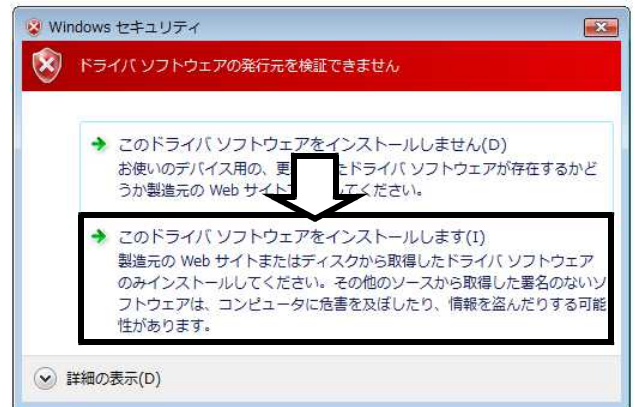
2 ”USB Programmer Software”のフォルダを開きます。

3 ”Driver”フォルダを開きます。

4 OSごとにフォルダがあります。ご使用のパソコンのOSにあったフォルダを開きます。
※WindowsVista,7に関しては32ビット版用と64ビット版用がありますので間違えないようご注意ください。

5 開くと実行ファイル、”USB18PRG～.exe”がありますのでダブルクリックして実行します。

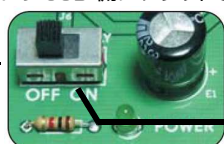
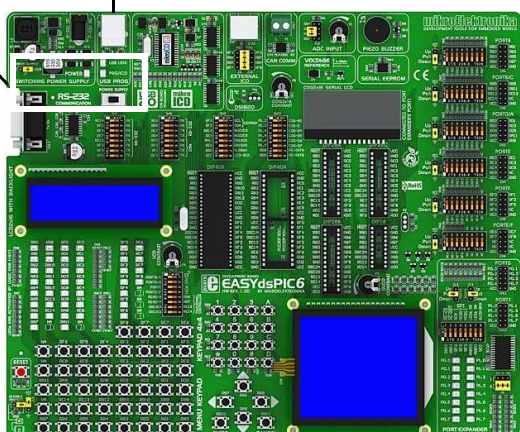
6 実行するとダイアログが表示されますので、そのまま”次へ”をクリックして続行します。

7 ドライバーのインストールが開始されます。
WindowsVista環境で下記のような警告ダイアログが表示された場合には、”このドライバソフトウエアをインストールします”をクリックして続行してください。

下図のようなメッセージが表示された場合には、"インストール"ボタンを押してください。

8 インストール作業が完了すると、下図のような画面になりますので、"完了"ボタンを押してインストールを完了します。
この時、"状態"のボックスに「使用できます」と表示されていることをご確認下さい。エラーが表示された場合には、ご使用OSの種類と、実行したデバイスドライバーの種類が一致しているかご確認下さい。

■パソコンと接続する

パソコンのUSBポートにdsPICF-400を接続します。

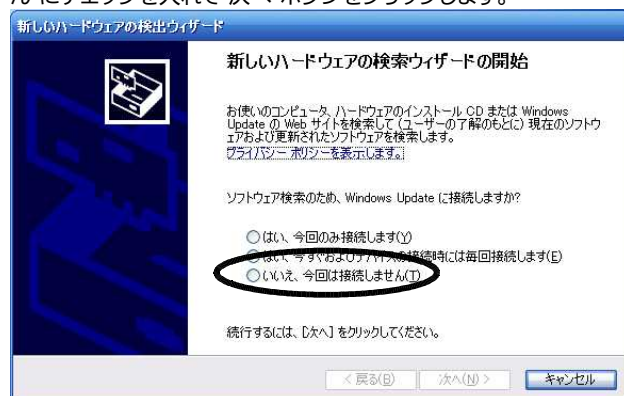
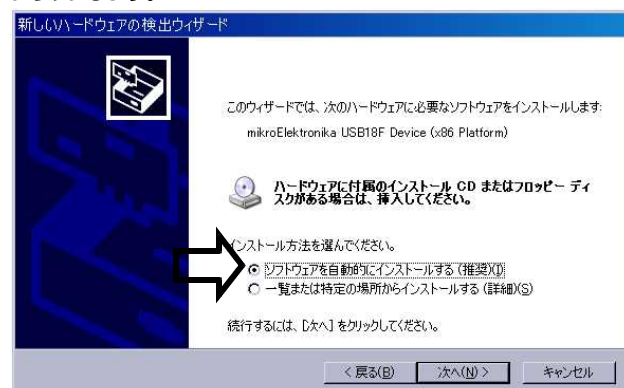
1 dsPICF-400の電源スイッチをONにします。
また、dsPICF-400の電源をUSBのバスパワーより給電しますので、J7の"USB"側にソケットを装着してください。

2 USBケーブルで、パソコンのUSBポートと、dsPICF-400を接続します。

※パソコンのUSBポートがUSB2.0対応の場合には高速書き込みが行えます。USB1.1ポートの場合には通常速度での書き込みとなります。

→接続するとPOWERの緑LEDが点灯します。
→新しいハードウエアの検出ウィザードが自動的に起動します。

以下OSによって操作方法が異なります。

■Windows2000, XPをご利用の場合

3 下図のようなダイアログが表示されたら"いいえ、今回は接続しません"にチェックを入れて"次へ"ボタンをクリックします。

4 下記のようなダイアログが表示されますので、"ソフトウエアを自動的にインストールする(推奨)"にチェックをいれて、"次へ"ボタンをクリックします。

5 「ソフトウエアをインストールしています。お待ち下さい」というメッセージが表示されます。しばらくするとインストールが完了して下図のような表示となりますので"OK"ボタンをクリックして終了します。

■WindowsVista,7をご利用の場合

3 WindowsVista,7の場合には、dsPICF-400を接続すると自動的にデバイスドライバーのセットアップが開始されます。
あらかじめデバイスドライバーは先の手順にてインストールされていますので、WindowsVistaは自動的にデバイスドライバーを検出してインストールを完了します。

接続後しばらく待ちますと、下図のようなポップアップがタスクバーに表示されます。

上図のホップアップが表示されればインストールは完了です。

■Windows98(SE), MEをご利用の場合

3 自動的にデバイスが検出されます。
"新しいハードウエアの追加ウィザード"が表示されたら「次へ」を押して続行します。

4 次のダイアログでは,"使用中のデバイスに最適なドライバを検索する"にチェックを入れて,「次へ」をクリックします。

5 "検索場所の指定"にチェックを入れて,「参照」ボタンを押します。
ドライバの場所を指定するダイアログが表示されますのでCD-ROM内の"USB Programmer Software"フォルダ内の"Driver"フォルダにある"WIN98_ME"フォルダを指定してください。CD-ROMドライブがQドライブの場合にはディレクトリは下記の通りとなります。
Q:¥USB Programmer Softeare¥Driver¥WIN98_ME

指定したら「次へ」を押して続行します。
インストールが完了したら,「完了」ボタンを押して終了します。

〜ここからは各OS共通の項目です〜

dsPICF-400ボード上の"USB Link"の黄LEDが点灯していることを確認してください。
黄LEDの点灯はPC側にdsPICF-400が正しく認識されていることを示します。

※"USB Link"のLEDが点灯していない場合には、正しくドライバーがインストールされていません。再度手順を確認の上、ドライバーをインストールし直しておためしください。

※デバイスマネージャーに"USB Devices by mikroElektronika"が追加されます。

WIndowsVista及び7環境でご使用のお客様へ ユーザーアカウント制御無効設定のお願い

WindowsVista及び7環境で、dsPICF-400をご使用の場合、WindowsIに搭載されたユーザーアカウント制御(UAC)機能がUSBポートへのアクセスを拒否することにより、下図のようなエラーメッセージが表示されて、書き込みができないという現象を確認しています。

この現象は、WindowsVista及び7において標準で有効になっているユーザーアカウント制御機能を無効に設定することで回避できます。下記の方法で設定を変更して頂けますようお願い致します。

※なおこのエラーは、WindowsVista及び7以外の環境でお使いのお客様には関係ありません。

■Windows Vistaのユーザーアカウント制御を無効に設定する

1 スタートボタンをクリックして、コントロールパネルを表示します。

2 コントロールパネルから"ユーザーアカウント"をダブルクリックします。"ユーザーアカウント"アイコンがない場合には、ウインドウ左上の"クラシック表示"をクリックして表示を変更します。

3 "ユーザーアカウント制御の有効化または無効化"のリンクをクリックします。

4 "ユーザーアカウント制御(UAC)を使ってコンピューターの保護に役立たせる"のチェックを外します。

5 "OK"ボタンを押して完了します。パソコンを再起動します。

■Windows7のユーザーアカウント制御を無効に設定する

1 スタートボタンをクリックして、コントロールパネルを表示します。

2 コントロールパネルから""ユーザーアカウントと家族のための安全設定"をクリックします。

3 "ユーザーアカウント"をクリックします。

4 "ユーザーアカウント制御設定の変更"をクリックします。

5 画面に表示されるスライドバーを一番下の"通知しない"に設定します。

6 "OK"ボタンを押して完了します。パソコンを再起動します。

ソフトウエアのインストール

開発に必要なソフトウエアをインストールします。なお、ソフトウエアをインストールする際には、必ずUSBポートからdsPICF-400を外した状態で行ってください。

■MPLABのインストール

MPLABはPICマイコンのプログラム開発用の統合開発環境です。PICマイコンの開発元マイクロチップ社から提供されています。MPLABがお手持ちのパソコンにインストールされていない場合には、インストールしてください。

1 CD-ROM内の"MPLAB"フォルダを開きます。
"SETUP.EXE"をダブルクリックして実行します。
"Next>"をクリックして続行します。

2 "License Agreement"のダイアログが表示されたら内容をよくご確認の上、同意する場合には"I accept the terms of the licence agreement"にチェックを入れて、"Next>"クリックして続行します。
なお同意しないとインストールは続行できません。

3 "setup type"の選択ダイアログではすべてのプログラムをインストールしますので"Complete"にチェックを入れて"Next>"をクリックします。

4 MPLABをインストールするディレクトリを指定します。
特に設定する必要がなければデフォルトのままにしておきます。
インストールするディレクトリを変更したい場合には,"Browse"ボタンを押してインストールディレクトリを設定します。
なお変更した場合には,インストールディレクトリをメモするなど忘れないようにしてください。
"Next>"をクリックして続行します。

5 以降のダイアログでは次のように進めます。

■Application Maestro License
→"I accept the terms of the license agreement"にチェックをいれて"Next>"をクリックします。

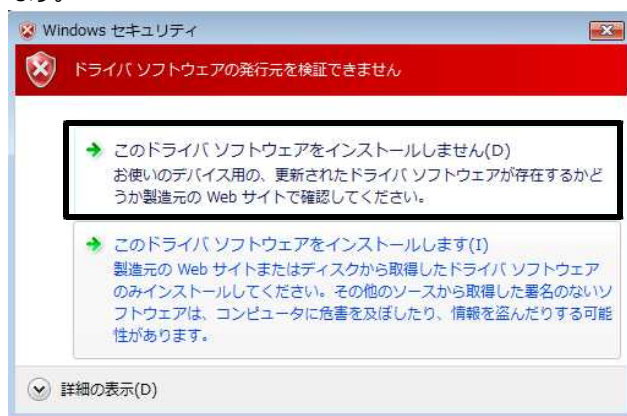
■MPLAB C32 License
→"I accept the terms of the license agreement"にチェックをいれて"Next>"をクリックします。

■Start copying files
→"Next>"を押します。
・・・インストールが開始されます。完了までしばらく待ちます。

6 インストールが完了すると、下図のようなダイアログが表示されます。MPLABバージョン8.00以降からはPICCコンパイラーのライトバージョンがMPLABに付属するようになったためです。
ここではOKボタンを押して続行します。

7 下図のようなPICC-Liteのインストール画面が表示されますので、ここでは"Cancel"ボタンを押してインストールをキャンセルします。

※PICC-Liteは、Hi-Tech社のPIC用Cコンパイラーです。MPLAB8.00以降ではライト版(機能限定版)が付属しました。インストールしていただいてもかまいませんが、本キットではこのソフトウエアは使用しません。

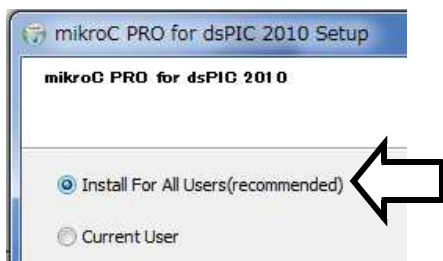
8 Windows Vista環境にインストールされている場合には、最後に下図のようなドライバーインストールに関するセキュリティ警告が表示されることがあります。
この場合には、"このドライバソフトウエアをインストールしません(D)"をクリックしてください。
ダイアログが消えたら"Finishボタンを押してインストールを完了します。

■mikroC PRO for dsPICのインストール

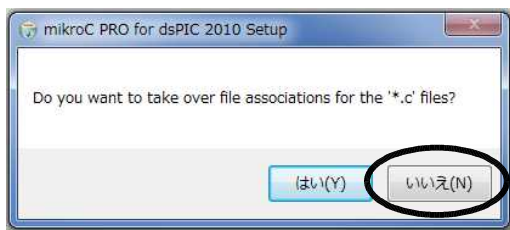
mikroElektronika社からリリースされているdsPICシリーズ用のANSI規格準拠のCコンパイラーです。

本dsPICF-400には、生成されるHEXファイルのコードが6KBまでに制限された体験版のCコンパイラーが付属しております。生成されるコードサイズに制限はありますが、使用できる機能及び期限等に制限はありません。この体験版Cコンパイラーをご使用になり、コードサイズの限定がない製品版をご希望の場合には、当方よりフル機能版を特別優待価格(10%引き価格)にて販売しております。

1 CD-ROM内の"Cコンパイラー"フォルダを開きます。
"SETUP.EXE"をダブルクリックして実行します。
"Next>"をクリックして続行します。

2 "License Agreement"のダイアログが表示されたら内容をよくご確認の上、同意する場合には"I accept the terms in the License Agreement"にチェックを入れて、"Next>"クリックして続行します。なお同意しないとインストールは続行できません。

3 次の画面では、"Install For All Users(recommended")にチェックを入れて、"Next>"ボタンを押して続行します。

4 "Choose Components"ダイアログが開きますので、"□Examples"にチェックが入っていることを確認して、"Next"ボタンを押します。

5 インストールするディレクトリを指定します。特に変更する理由がなければ、"Install"ボタンをクリックします。インストールが開始されます。インストール開始後下図のようなメッセージが表示されたら、"いいえ"をクリックして続行します。

6 "Completing the mikroC dsPIC Setup Wizard"というダイアログが表示されたら、"Finish"ボタンを押して終了します。
続いて、下記のようなメッセージが表示されますので、"はい"をクリックします。

7 再度、インストーラーが起動します。画面の指示に従って、インストールを進めてください。"License Agreement"のダイアログが表示されたら内容をよくご確認の上、同意する場合には"I accept the terms in the License Agreement"にチェックを入れて、"Next>"クリックして続行します。

8 インストールを完了すると再度、下図のようなメッセージが表示されます。"はい"を選択して、デバイスドライバーもハードディスクに配置しておきます。

■PICプログラマ用のソフトウエア

PICへプログラムを書き込む際に使用するプログラマソフトウエアです。mikroCをインストールした際にはmikroProg Suite For PICとして書き込みソフトウエアはインストールされていますが、mikroCをインストールしなかった場合には、書き込みソフトウエア単体でのインストールが必要です。
mikroCをインストールした場合には、この手順は飛ばしてください。

1 CD-ROM内の"USB Programmer Software"内にある"SETUP.EXE"をダブルクリックして実行します。インストーラーが起動しますので、"Next"をクリックして続行します。

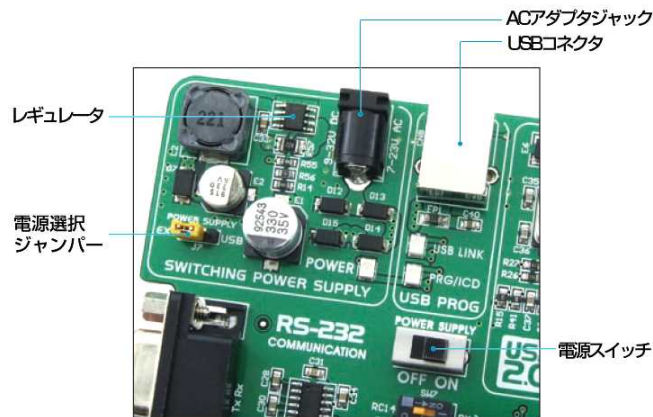
2 ライセンス許諾画面が表示されますので、同意される場合には、"I accept the terms in the License Agreement"の方にチェックを入れて、"Next"ボタンをクリックします。

3 次の画面はそのまま"Next"をクリックします。

4 インストールするディレクトリを指定します。mikroCをご使用になる場合には、必ずmikroCをインストールしたドライブと同じドライブにインストールしてください。(例えばmikroCをCドライブにインストールした場合には、こちらのソフトウエアもCドライブにインストールしてください。)
"Install"ボタンを押すとインストールが開始されます。

5 インストールが完了したら"Finish"ボタンを押して完了します。

電源の投入方法

dsPICF-400は、USB接続にてUSBバスから電源の供給を受けることができます。この場合、別途電源を供給する必要はありません。
しかしdsPICF-400側でおよそ300mA以上を消費する予定がある場合にはUSBバスパワー給電では電力が不足する場合があります。この場合には別途外部からACアダプタにて電源を供給する必要があります。給電方法はジャンパーソケットJ7によって設定を行います。
なお工場出荷時の設定はUSBバスパワー給電の設定になっています。

■USBバスパワーから給電する場合
dsPICF-400ボード上の"SUPPLY SELECT"ジャンパーピン(J7)を「USB」側にショートします。

■ACアダプタから給電する場合
dsPICF-400ボード上の"SUPPLY SELECT"ジャンパーピン(J7)を「EXT」側にショートします。

※ACアダプタは出力電圧がDC9Vで500mA以上のものをご用意ください。径は2.1φです。(当方にて専用ACアダプタを販売中です)

■電源スイッチ
dsPICF-400には、電源スイッチが装着されています。このスイッチは、給電方法に関係なく、ボードのスイッチのON/OFFができます。

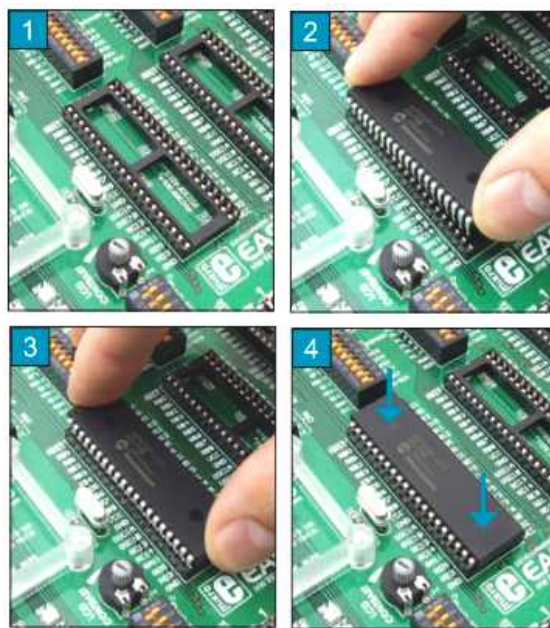
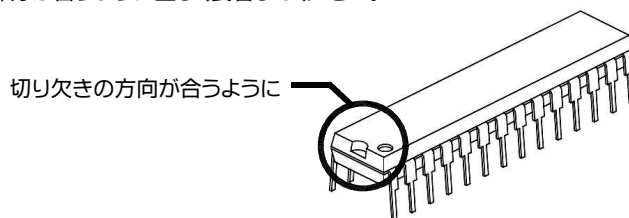
対応デバイス

dsPIC30F1010 , dsPIC30F2010 , dsPIC30F2011
dsPIC30F2012 , dsPIC30F2020 , dsPIC30F2023
dsPIC30F3010 , dsPIC30F3011 , dsPIC30F3012
dsPIC30F3013 , dsPIC30F3014 , dsPIC30F4011
dsPIC30F4012 , dsPIC30F4013

※上記のデバイスのDIPタイプのデバイスに対応しています。

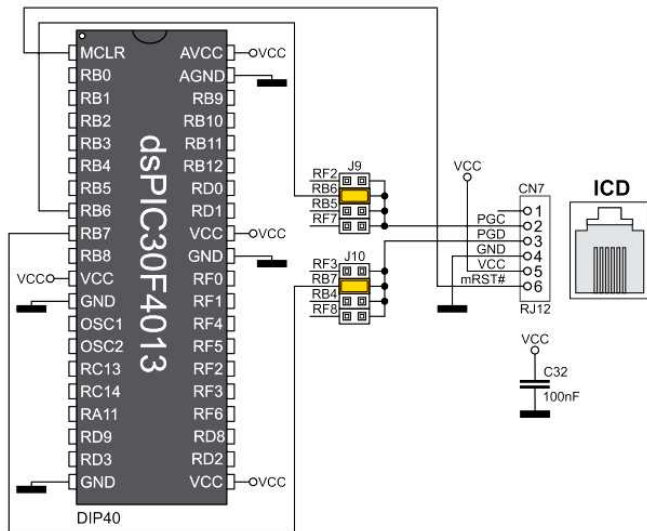
PICマイコンの装着

使用するPICマイコンを、dsPICF-400ボード上のICソケットに装着します。工場出荷時は、dsPIC30F4013-30I/Pがすでに装着されています。

装着できるPICマイコンは、必ず1種類だけです。複数のソケットにマイコンを装着すると故障の原因となります。装着時には、装着方向を間違わないようにICソケットの切り欠き部分とdsPICマイコンの切り欠き部分が合うように正しく装着してください。

■40ピン及び28ピンデバイスを装着する場合
40ピン及び28ピンのdsPIC30Fシリーズのマイコンでは、型式によってピンの配置が異なっています。
dsPICF-400では、40ピンソケットを2つと、28ピンソケットを3つ用意しており、ピン配置の異なるデバイスでも装着できるように設計されています。

基板上に装着できるデバイスの型式がシルク印刷されていますので、装着するデバイスの型式に応じて適切なソケットにデバイスを装着してください。その他、シルク印刷にない型式のデバイスの場合には、装着するデバイスのピン配置が、dsPICF-400基板上のピン配置の印刷と一致するソケットに装着してください。デバイスのピン配置はデータシートで確認できます。

MPLAB-ICD2及びICD3を使用する場合

dsPICF-400には、マイクロチップ社純正のMPLAB-ICD2又は、MPLAB-ICD3を接続するための6ピンモジュラーコネクターが付いています。ここにMPLAB-ICD2又はICD3を接続することでdsPICF-400をターゲットボードとして使用することができます。
内部の結線は下記のようになっています。

PGCピンと、PGDピンの結線は、使用するデバイスに応じてジャンパーソケットJ10及びJ11にて設定できます。デバイスのデータシートを参照し、ジャンパーソケットにて適切なピンに設定してください。

■MPLAB-ICD2及びICD3の設定

MPLAB-ICD2・ICD3のVddピンはdsPICF-400のVddと接続されています。MPLAB-ICD2・ICD3接続して、dsPICF-400をターゲットボードとして使用する場合には、電源の設定方法及びMPLAB-ICD2・ICD3の電源設定を下記のように設定してください。接続及び設定を誤ると、機器が故障する原因となります。

・dsPICF-400の電源は必ずACアダプタにて外部電源から給電する
・dsPICF-400のUSBはパソコンと接続しない
・dsPICF-400のUSBライター機能は使用しない
・MPLABで電源の設定を、"Target has own power supply"にする

ICD2・ICD3では、ターゲットボード側にICD2・ICD3側から電源を供給するモードと、ターゲットボードとICD2・ICD3の電源を別々にするモード、2つのモードが選択できます。ICD2・ICD3を利用する場合には、必ず後者のターゲットボードとICD2・ICD3の電源を別々のするようにします。(dsPICF-400の電源給電はdsPICF-400に接続したACアダプタで行い、ICD2・ICD3の電源給電は、USBバスパワーで行う)
下記の手順で電源の設定を行ってください。

1 dsPICF-400とICD2・ICD3は接続していない状態で、MPLABを起動し、メニューバーの"Debugger"又は"Programmer"をクリックして、使用するツールとして"MPLAB ICD2"又は"MPLAB ICD3"を選択し使用できるようにします。

2 再度メニューバーの"Debugger"又は"Programmer"をクリックして、"Settings"をクリックします。

3 表示されたダイアログの中で"Power"タブをクリックします。

4 "Power target circuit from MPLAB ICD2(又はICD3)"のチェックを外します。無効になっている場合には、チェックが入っていないことを確認してOKボタンを押します。

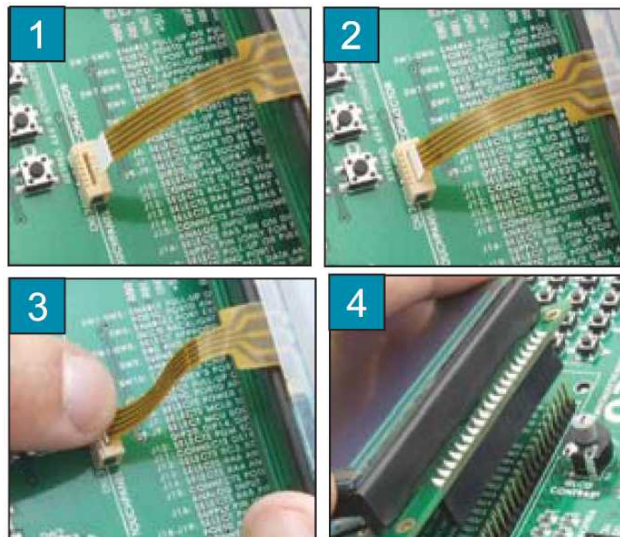
5 dsPICF-400に電源を投入し、J10及びJ11のジャンパーでPGD及びPGCのピンを設定します。
ICD2・ICD3の電源が入っていることを確認して、6ピンモジュラーケーブルで、dsPICF-400のモジュラーソケットに接続します。

設定を正しく行ってご使用ください。MPLAB ICD2又はICD3を使用する場合には、絶対にdsPICF-400に搭載のUSBライター機能は使用しないでください。dsPICF-400に搭載のUSBライター機能を使用すると、両製品が破損します。この誤りを防ぐため、MPLAB ICD2又はICD3を使用する場合には、dsPICF-400のUSBポートには何も接続せず、給電は必ずACアダプタから行うようにしてください。

GLCDを取り付ける場合

dsPICF-400には、128×64ドットのグラフィックLCDを取り付けることができます。また4線式抵抗膜方式のタッチパネルを搭載したGLCDを取り付ければ、基板上のフラットケーブルコネクタにタッチパネルの信号線を取り付け、タッチパネルコントローラに信号を接続することができます。

1 GLCDを取り付ける場合には、必ずdsPICF-400の電源を切断してください。

2 タッチパネル付きのGLCDの場合には、GLCDを取り付ける前に、タッチパネルのフラットケーブルを基板上のコネクタに押し込んで装着します。コネクタは抜けないように堅い部分をコネクタに押し込んでください。

3 続いて128×64ドットGLCDをソケットに真っ直ぐ差し込みます。

※GLCDの脱着時は必ずdsPICF-400の電源を切断した上で作業を行ってください。

dsPICF-400の各部の説明

①外部給電用ACアダプタジャック
③USBライター部
⑳温度センサIC DS18S20用コネクタ
⑲CAN通信用コネクタ
⑱ADコンバータ評価用ボリュームと基準電圧生成部
⑰圧電ブザー
②給電方法選択ジャンパー
⑯実験用シリアルEEPROM
④RS232Cポート
⑮SPI通信駆動16文字2行LCD
⑤16文字2行キャラクタLCD
⑭全I/Oピン外部取出ヘッダーピン及びプルアップ/ダウン設定スイッチ
⑥41個のピン状態表示用LED
⑬SPI通信実験用ポート拡張回路部
⑦リセットスイッチ
⑧ピンショート防止ジャンパー(J11)
⑨アクティブLow/High設定ジャンパー(J15)
⑩41個のタクトスイッチ
⑪キーパットとマトリックススイッチ
⑫GLCD及びタッチパネル用フラットケーブル取付コネクタ

①ACアダプタジャック及び電源セレクトジャンパ

dsPICF-400の電源をACアダプタから給電する場合には、このジャックにDC9V出力で2.1φのACアダプタを接続します。センター極性は問いません。電流は500mA以上を取り出せる安定化されたものを使用してください。(センター極性は問いません)
※給電方法については、②のジャンパー設定が必要です。

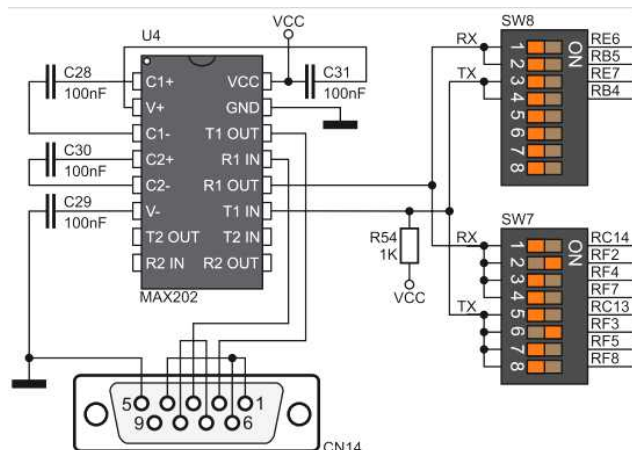
②給電方法選択ジャンパー(J7)

dsPICF-400への給電方法を選択します。USBバスパワーから給電するのか、ACアダプタより給電するのかをジャンパーピンで設定します。

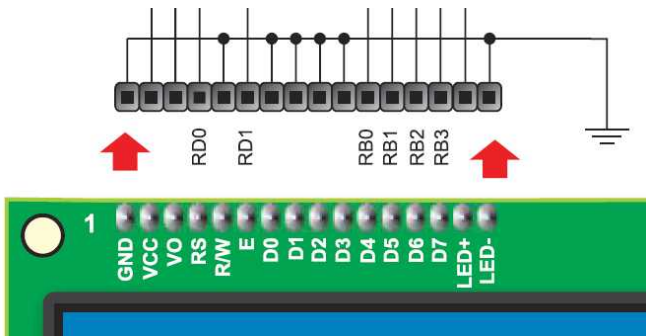
③オンボードUSBライター、USBコネクタ

オンボードUSB2.0マイコンライター部です。
パソコン本体のUSBポートに接続します。付属の専用CコンパイラーmikroCと組み合わせることで、ICD機能を実現することができます。
USBデバイスとして正しくパソコンに認識されると、USBコネクタすぐ下の"USB LINK"のLEDが点灯します。プログラム書き込み中やICD実行中は"PRG/ICD"のLEDが点灯します。

④RS232C通信ポート

レベル変換IC搭載のRS232Cポートです。
RX(受信データ)及びTX(送信データ)の信号線の接続先は、SW7及びSW8のディップスイッチによって、選択することができます。
RS232CのTX線とRX線をそれぞれdsPICのどのピンと接続するかは、SW7・SW8部に記載されている基板のシルク印刷を見ながら設定を行います。
RS232C通信の実験をする場合には、パソコンとD-SUB9ピンの全結線ストレートケーブルをご使用下さい。

⑤16文字2行キャラクタ液晶ディスプレイ

2行16桁のLCDです。データ線にD7～D4の4ビットを使用する4ビットモードで使用できます。コントラスト調節は、P4ボリュームで行うことができます。

LCDのデータピンD4～D7はそれぞれdsPICマイコンのRB0～RB3に、EnableビットはRD1に、RSピンはRD0に接続されています。

なお、RB0～RB3及びRD0とRD1はLCDの各ピンと接続されている関係上、LCDを取り付けた状態で、dsPICからこのピンをI/Oポートとして使用しようとすると、電圧レベルが正しく0V-Vccで出力されません。このピンをI/Oポートとして使用する場合には、LCDを物理的に取り外してご使用ください。

⑥41個のI/Oピン状態表示LED群

各ビットの状態を表示できるLED群です。

該当のピンがHレベルになると点灯します。本製品で使用されているLEDの順方向電流は、1mAです。抵抗器を通してマイコンのピンと接続されています。

LEDのカソード側をGNDと接続するかどうかを、SW12のディップスイッチによってポート単位で設定できます。

基板上にスイッチ横のシルク印刷に、どのポートのスイッチかが記載されています。dsPIC30Fマイコンの場合、ポートの配置が変則的ですので効率よく使用できるようにスイッチの配置が設計されています。例えば、付属しているdsPIC30F4013の場合、PORTDとPORTFはありますが、PORTEは存在しません。そのような変則的な設計となっているため、スイッチについても別のポートと共有するような設計となっています。

マイコンのI/OピンにLEDが物理的に接続されていると、LED側に電流が流れ電圧降下が発生します。よって、ピンをI/Oピンとして使用する場合、LEDと接続されていると電圧降下により、Hレベル時にVcc電圧となりません。H-LにてVp-pをVcc-GND電圧で使用したい場合にはSW7の該当ポートのスイッチをOFF側に設定して、物理的にLEDからI/Oピンを切り離してください。

⑦リセットスイッチ

マイコンの$\overline{\mathrm{MCLR}}$ピンと接続されており、スイッチを押すとdsPICマイコンの$\overline{\mathrm{MCLR}}$ピンがGND電位となり、ハードウエアリセットがかかります。

⑧ピンショート防止ジャンパー(J11)

J11のジャンパーは、短絡時保護用抵抗を無効にするか、有効にするかを設定するものです。下図の通り、J11は220Ωの抵抗に並列に接続されており、ジャンパーソケットを外すと、タクトスイッチは220Ωの抵抗を介してVcc又はGNDと接続されます。

これによって例えば、dsPICマイコン側が意図せず出力設定になっていて、あるロジック状態になった際、誤ってタクトスイッチを押した時に大きな電流が流れずに220Ωの抵抗を介して電流が流れ、dsPICマイコンの破損を防止できます。

通常使用時は、J11のジャンパーは外しておくことを推奨します。

⑨アクティブLow/High選択用ジャンパ(J15)

⑩のタクトスイッチを押した時、該当のピンをLowにするのか、HighにするのかをJ設定するためのジャンパーです。

スイッチを押したときにLowになるものをアクティブLow、逆にスイッチを押したときにHighになるものをアクティブアクティブHighとよびます。J15のジャンパをVcc側にするとアクティブHighに、GND側に設定すると、アクティブLowとなります。

ただし、押しボタンスイッチの設定は、J15のアクティブLow/High設定と、J1～J6のプルアップ/プルダウンの設定を適切に組み合わせる必要があります。J15をアクティブLow設定にした場合には、該当する押しボタンスイッチの接続されたポートとピンは、プルアップしておく必要があります。下記に例を示します。

J1～J6のジャンパで、各ポートのプルアップ/プルダウン設定ができ、SW1～SW6で各ビットに対してその設定を有効にするか、無効にするかを設定できます。スイッチの論理は、このプルアップ/ダウンの設定と、J15の設定とが正しい組み合わせでないと正しく動作しませ

んのでご注意ください。先の図の設定状態にすることでスイッチを押さない時は、該当ピン(RD0)はHレベル、スイッチを押した時、Lレベルになるようになります。

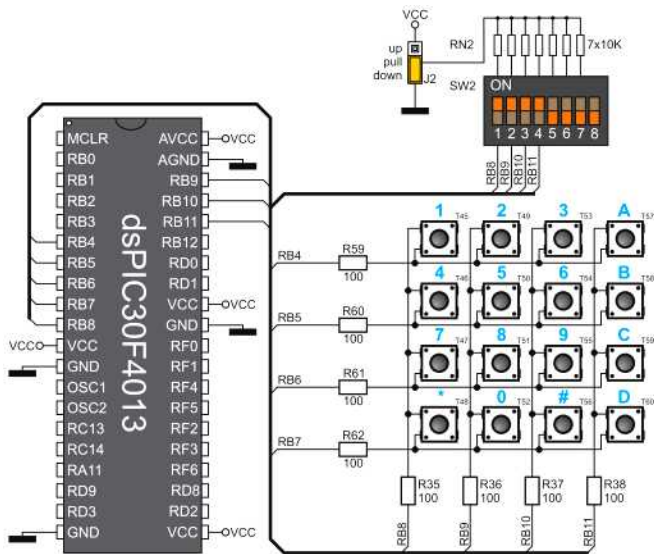
⑩41個I/Oピン接続タクトスイッチ群

J15のジャンパー設定によって、アクティブHigh又はアクティブLowが選択可能な41個のタクトスイッチです。

タクトスイッチ群の左側にあるJ15をVccと刻印された側に設定すると、アクティブHighに、GNDと刻印された側に設定すると、アクティブLowになります。(詳しくは⑨の項目をご参照ください)

⑪キーパッドとマトリックススイッチ

4×4のマトリックス配列のキーパッドです。PORTBのピンを8ビット分使用してマトリックスを構成しています。

回路構成は下図の通り、行と列から構成されています。

一般的な使い方としては、RB8～RB11の列については、ディップスイッチSW2と、J2ジャンパーの設定でプルダウンの設定にしておきます。よって、この時何もせずにRB4～RB7の行の状態をスキャンすると、すべて論理は0(Lレベル)となります。

ここで、行のRB4～RB7について1つずつ順番にある一定間隔で論理を1(レベルをH)にして、それを繰り返します。この時に列のRB8～RB11の状態をスキャンします。

例えば、RB4が1の時に"スイッチ3"が押されればRB10の論理が1となり"スイッチ3"が押されたことを特定できます。

RB4～RB7の順番に1にする繰り返し周期は数ミリ秒とすることで、どの行が1の時、どの列が1になったかを組み合わせることで、16個のマトリックスのうち1つのキーを特定することができます。

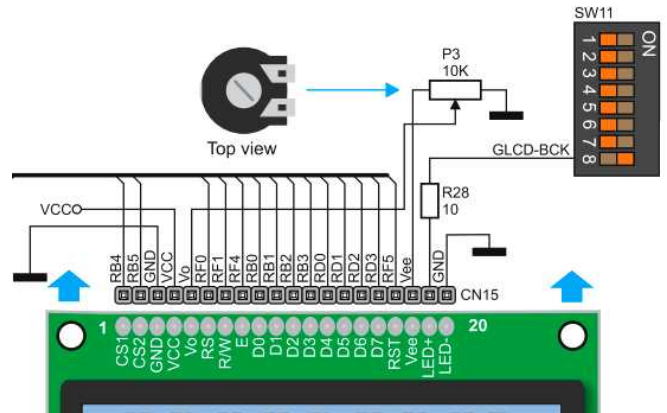
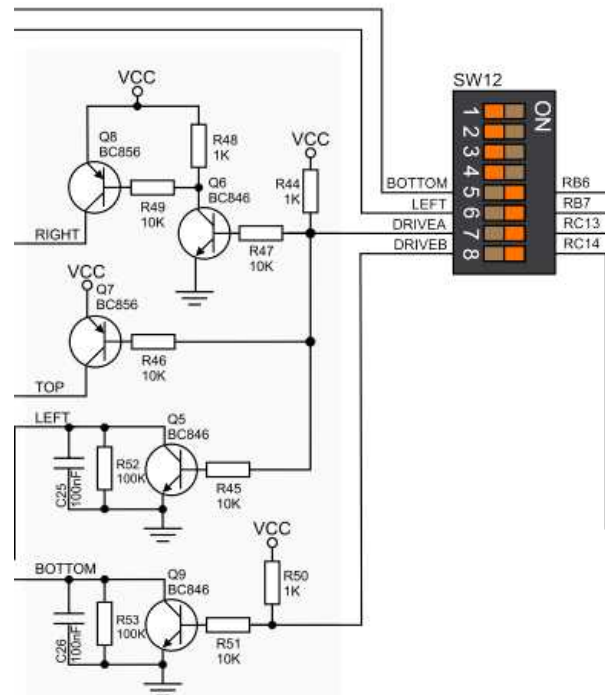
キーマトリックスの仕組みを使えば16個のキーがあっても8つのピンを消費するたけで、キーを特定できるメリットがあります。

4×4マトリックススイッチの下にあるのは、キーパッドです。シルク印刷にある通りのピンにそれぞれ接続されています。アクティブLow/Highの設定は、タクトスイッチ群と同じくJ15で設定できます。

⑫GLCD及びタッチパネル用フラットケーブル取付コネクタ

128×64ドットのグラフィックLCDを接続するピンです。

LCDのコントラストは、P3のボリュームで調整します。また、バックライト付きのLCDの場合にはディップスイッチSW11の8番スイッチにて、ON/OFFの設定が可能です。

結線は下図の通りです。

なお、この部分には4線式抵抗膜方式のタッチパネルの付いたGLCDも取り付けられるようになっています。本GLCD取り付け部左側には、タッチパネル用の4芯フラットケーブル用のコネクタがあります。タッチパネルの信号はタッチパネルコントローラ回路部に接続されます。下にその回路部を示します。

SW9の5番～8番スイッチで、タッチパネルコントローラ回路とdsPICを物理的に接続するかどうかについて設定できます。

付属のmikroCでは組込関数によって簡単にこのコントローラ回路を使って、タッチパネルの座標位置を取得できるようになっています。

なお、タッチパネルを使用しない場合にはこのスイッチは必ずOFFにしておく必要があります。

⑬SPI通信実験用ポート拡張回路部

マイクロチップ社のSPI通信ポート拡張IC、MCP23S17を搭載した、拡張ポート回路部です。

SPI通信によって駆動できる16ビットのI/Oを増設することができます。拡張された16ビットのI/OにはそれぞれLEDが接続されており、ピン状態のH-Lを視認できます。SPI通信の実験や学習に最適です。

各I/Oピンは外部に取り出せるようヘッダーピンが付いていますので、ポートが足りない時に使用することができます。

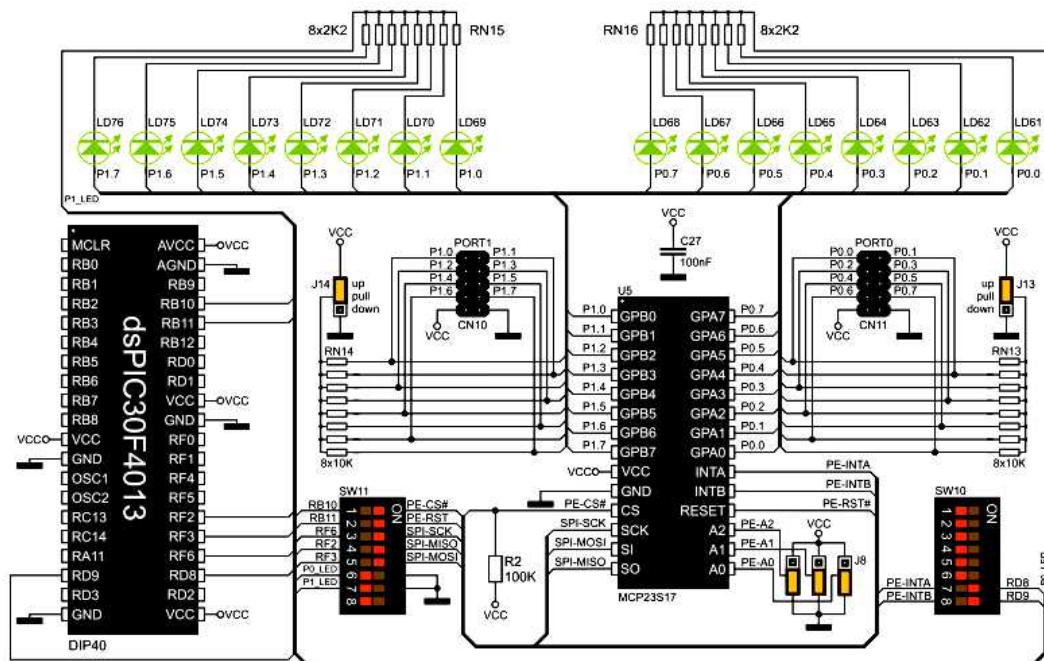
ディップスイッチSW11によって、SPI通信の各信号線をdsPICマイコンと接続するかを選択できます。使用する場合には、SW11の1～5をONにします。また、LEDについて点灯させる場合には、SW11の6と7で設定します。

割込ピンのINTAとINTBの出力は、ディップスイッチSW10の7と8を介してdsPICマイコンと接続されます。必要に応じて設定します。

ポート拡張IC、MCP23S17のポート出力は、LEDと接続されている他、CN10とCN10の10ピンヘッダーコネクタから取り出すことができます。ピンアサインは基板のシルク印刷をご覧ください。

また、J13とJ14ジャンパーによって、各ポート全体のプルアップ又はプルダウンを設定できます。

J8のジャンパー設定は、MCP23S17のデバイスアドレスを決める3ビットのピンです。通常は000として3つのソケットをすべて0側にセットしておきます。

※MCP23S17の使い方はMCP23S17のデータシートをご覧ください。

⑭全ビット取り出し可能ヘッダピン
各ビット個別プルアップ/プルダウン有効・無効設定スイッチ

全I/Oピンをこの部分のヘッダピンから取り出せます。

ポート単位でプルアップ/プルダウンがJ1～J6のジャンパーによって設定できます。プルアップに設定すると、10KΩの抵抗器によって、+5V(Vcc)へプルアップされます。プルダウンに設定すると、同様に、10KΩの抵抗器によってGNDへプルダウンされます。

また、J1～J6のフルアップ/プルダウン設定は各ポートにおいて、各ビットごとにディップスイッチSW1～SW6によって個別に有効にするか、無効にするかの設定ができます。

ディップスイッチの部分のシルク印刷に従ってピンの設定を行ってください。

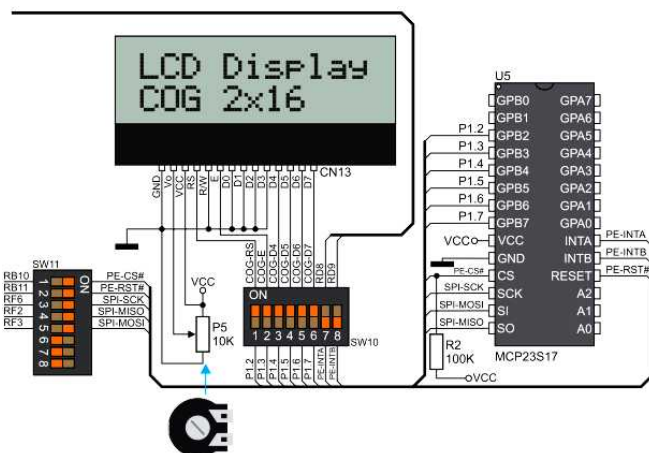
⑮SPI制御オンボード2×16LCDディスプレイ

SPI通信によって、表示を行うことのできる2行16文字のオンボードLCDです。

⑬のSPIポート拡張機能を使用して、dsPICマイコンからはSPI通信で表示させたい文字列を送ります。SPI通信の実験に最適です。

本ディスプレイを使用する場合には、ディップスイッチSW10をすべてONに設定します。これにより、LCDの各制御線がSPI通信ポート拡張回路部のMCP23S17のポート1に接続されます。

また、dsPICマイコンとSPI通信ポート拡張回路部の間でSPI通信を行うために、SW11の1～5をON側に設定し、物理的に信号線をdsPICマイコンと接続します。

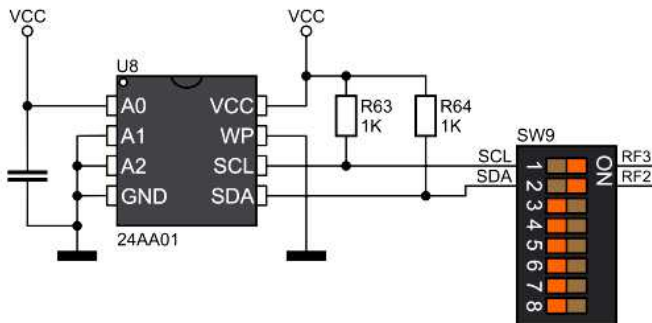
SPI通信でLCDの駆動ができますので、SPI通信の実験や学習に最適です。

コントラストはP5のボリュームで調整できます。

使用しない場合には、SW10及びSW11の1番～5番スイッチはOFFの状態にしておきます。

⑯実験用シリアルEEPROM

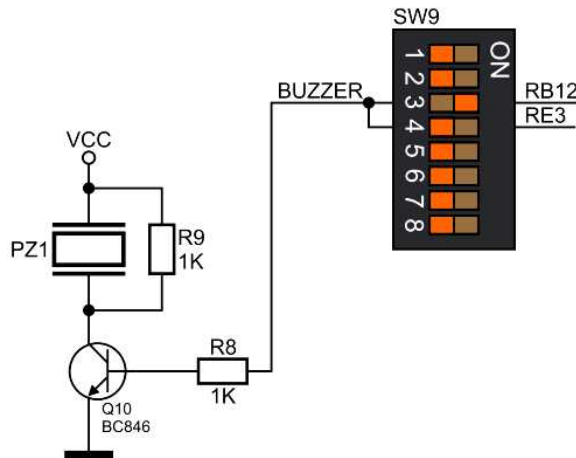
I2Cインターフェイス実験用に使用できるI2Cインターフェイス式の不揮発性メモリー(EEPROM)、24AA01(最大1kbitデータ)が接続されています。

I2C通信での通信実験やEEPROMへのデータの書き込み、読み出しの実験に使用できます。

I2CのSDA及びSCLの結線は、ディップスイッチSW9によって、dsPICのピンと接続するかを指定できます。SDAはRF2と、SCLはRF3と接続することができます。なお両信号線は、1KΩの抵抗器によって、プルアップされています。

I2CのスレーブアドレスはA0〜A2ピンの3ビットで決まり、配線がA0のみVccに接続されているため、"001"になっています。よって、スレーブアドレスは1になります。

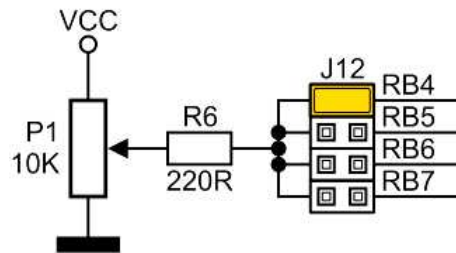
⑰圧電ブザー

実験用の圧電ブザーです。NPN型トランジスタによつdsPICの出力が増幅されて圧電ブザーを駆動します。

SW9によってトランジスタのベースをRB12又はRE3のどちらに接続するかを設定できます。

本圧電ブザーは他励式のため、発振した信号を作り、その信号を印加することで音として発音します。人間の聴覚では20Hz〜20KHz程度までを認識できますが、ブザー音として使用する場合に適当な周波数は2KHz〜4KHz程度です。

⑱ADコンバータ評価用ボリュームと基準電圧生成部

ADコンバーター評価用として、P1ボリュームが実装されています。ボリュームは10KΩで0V〜5Vの電圧を左隣のJ12ジャンパーによって設定したdsPICのピンに印加できます。

J12で印加するピンを選択します。ADコンバータを使用しない場合にはジャンパーソケットは必ず外しておいてください。

RB0を基準電圧のVref+として使用する場合、dsPICF-400のボード内で生成した基準電圧の4.096VをRB0に印加させることが可能です。基準電位を4.096Vにさせると、10ビット分解能のADの場合、1ビットあたり4mVとして計測できます。

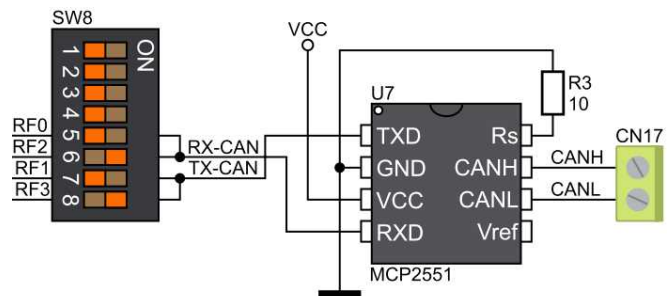
基準電圧はレギュレターによって生成され、RB0にその電圧を印加するかとどうかをSW9の7番スイッチで設定できます。このスイッチをONにすると、4.096VがRB0に印加されます。

【注意!!】

P1の可変抵抗器では、指定したピンに対して0V〜5Vまでの電圧を印加できますが、誤ってdsPICのピン設定を出力設定したピンに電圧を印加してしまうと、大きな電流が流れて素子を破損したり、ボード本体が破損したりすることがあります。使用する場合には、プログラムによるピンの設定及び、物理的な配線が短絡するような条件にならないよう十分ご注意下さい。

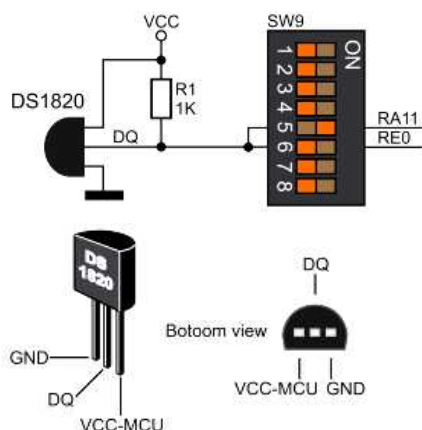
⑲端子台付きCANコントローラ部

CAN通信用のコントローラ、MCP2551を搭載したCAN通信実験用回路部です。CN17にはCAN-HとCAN-Lの信号線が接続されています。

MCP2551のRXピン及びTXピンの接続は、SW8のスイッチ5〜8で設定することができます。CAN通信実験をする場合には、適宜SW8を設定してご使用ください。使用しない場合には、スイッチをOFFにしておきます。

⑳温度センサIC(DS18S20)用コネクタ

ワンワイヤ通信方式を採用したダラスセミコンダクタ社(現マキシム社)のワンチップデジタル温度センサIC、DS18S20を接続するためのコネクタです。

DS18S20は非常にシンプルな構成で−55℃〜125℃までを計測できるワンチップ温度センサICです。当方でオプション品として販売しています。mikroCの組込関数から簡単に使用できます。

SW9のスイッチ5及び6にて、DS18S20のDQ線をdsPICのどのピンと接続するかを指定できます。DQピンは1KΩの抵抗でプルアップされています。使用する場合には、ソケットにDS18S20を極性を間違えないように装着し、SW9のスイッチ5、6を適宜設定してご使用ください。

外部発振子の取り付け

dsPICF−400では、外部に水晶発振子やセラミック発振子を取り付けてクロックをdsPICマイコンに供給できます。標準では10MHzの水晶発振子がX1ソケットに取り付けられています。

発振子を装着するソケットはX1と、X2の2つがあり、接続されているICソケットが異なっています。下記の通りとなっています。

■X1のソケット

DIP28A、DIP40A、DIP40BのICソケットに接続されています。

■X2のソケット

DIP28B、DIP18、DIP28CのICソケットに接続されています。

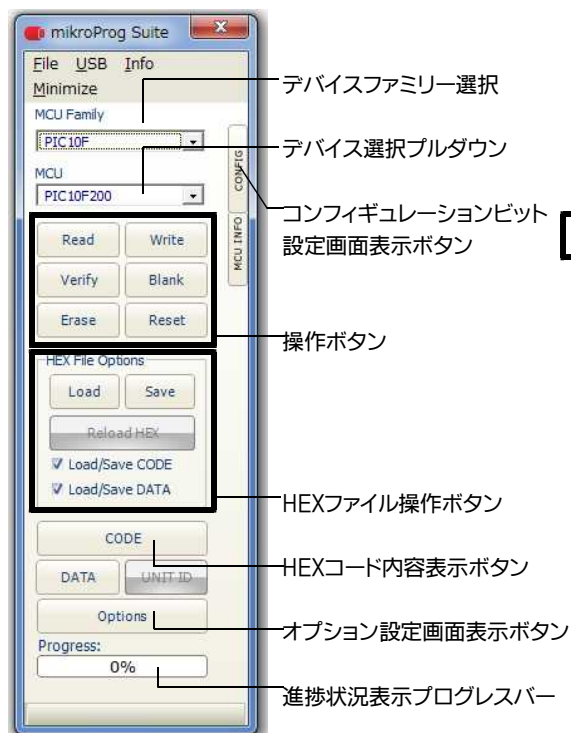
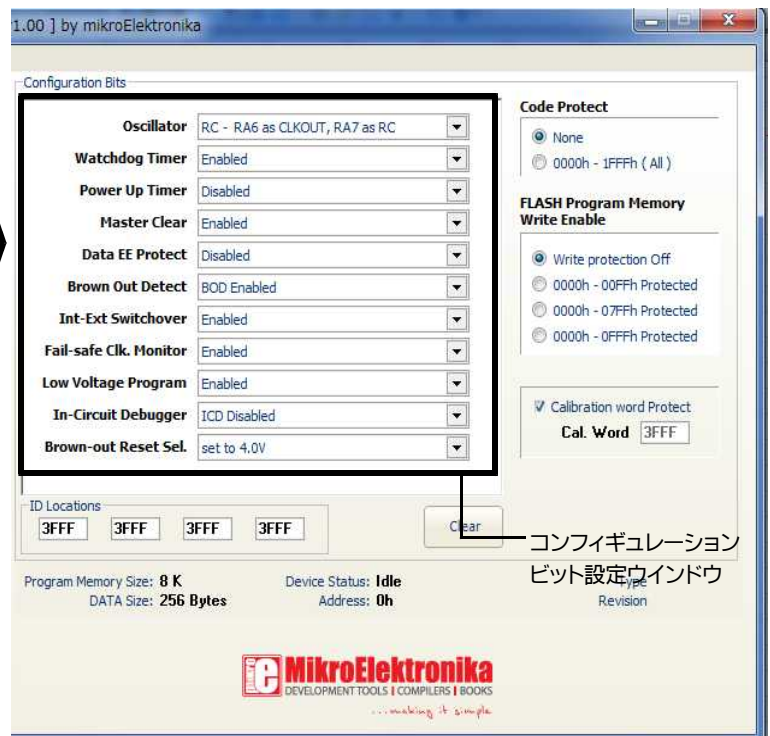
使用するデバイス及び装着しているICソケットに応じて発振子を正しく装着してご使用ください。

書き込みソフトウエアの使用方法

付属のUSB2.0マイコンライターの使い方を紹介します。
マイコンライターは、コンパイラで作成したHEXファイルをPICマイコンへ書き込む際に使用するソフトウエアです。
ソフトウエアは、mikroProg Suite for PICを使用します。付属のCコンパイラー、mikroCを使用すると、コンパイルと書き込みを連動させて行うことができて大変便利です。

■PICプログラマーの起動
PICプログラマーは、Windowsのスタートメニュー又はデスクトップに作成されたショートカットから起動できます。

スタートメニューから起動する場合には、"スタート"→"プログラム"→"Mikroelektronika"→"mikroProg Suite For PIC"→"mikroProg Suite For PIC"の順でクリックして起動します。

■アプケーション概要

初期画面

"CONFIG"ボタンを押した時に右側に表示される画面

■操作の順番
次の順番でHEXファイルを書き込みます。

1 デバイス設定プルダウンより、書き込むデバイスを選択します。
最初に"デバイスファミリー選択"プルダウンからデバイスファミリーを選択します。"dsPIC30F"が選択できます。

2 続いて"デバイス選択プルダウン"より書き込みを行うデバイスを選びます。

3 書き込むHEXファイルを読み込みます。
HEXファイル操作ボタンの"Load"ボタンを押します。
ダイアログが表示されますので、ファイルを選択します。
→ファイルが読み込まれるとステータスバーに、ファイル名が表示されます。

※同じファイルを再度ロードする場合には、いちいち上記のようにファイルを指定しなくても、"Reload HEX"ボタンを押すことで再読込ができます。

※ファイル名や、ファイルの配置してあるディレクトリ名に日本語や2バイト文字が含まれている場合、正しくHEXファイルの読み込みができません。必ず半角英数字になるようご注意ください。

4 一般的にはこれで書き込み準備は完了ですが、必要に応じてコンフィギュレーションレジスタの設定を確認します。
コンフィギュレーションレジスタは、プログラムを書き込む時にだけしか設定できないPICマイコン全体の動作や基本的な設定などを行う特殊なレジスタです。
コンフィギュレーションビットの設定が間違っているとプログラムや回路が正しくてもプログラムは動作しません。通常、コンフィギュレーションビットの設定内容は、プログラム内に埋め込まれているため、HEXファイルを読み込んだ時点で自動的に設定されます。
しかし、設定が正しくないとプログラムの動作に問題が発生することがありますので、書き込み前に確認されることをお奨めします。

"コンフィギュレーションビット設定画面表示ボタン"をクリックします。右側に設定ウインドウが展開します。

5 必要に応じてコード保護機能も設定します。
コードプロテクションを有効にすると、一度書き込んだHEXデータは読み出せなくなります。コードを不正にコピーされることを防止できます。通常は、"None"及び"Write protection Off"に設定します。

6 設定が完了、確認したら書き込みを行います。
PICD-32MX2本体の"USB LINK"の黄LEDが点灯していることを確認してから操作ボタンの"Write"ボタンをクリックします。
書き込みを開始します。
書き込み動作中は、進捗表示プログレスバーに進捗状況が表示されます。完了するまで待ちます。

書き込みが完了すると、プログラムがすぐにボード上で動作を開始します。

■その他の機能

・Read機能

→PICマイコンからデータを読み込みます。
読み込んだデータは、"Code"ボタンで閲覧できます。またメニューバーの"File"→"Save HEX"で保存できます。

・Verify機能

→現在読み込まれているHEXファイルの内容と、PICに書き込まれているプログラムの内容が一致しているか検証します。

・Blankチェック機能

→現在装着されているデバイスのプログラムメモリがブランク(空)かチェックします。

・Erase機能

→現在装着されているデバイスのプログラムメモリの内容を消去します。

主な仕様

電源電圧:	USBバスパワー給電時 DC5V
	ACアダプタ給電時 DC9V
給電方法:	USBバスパワー又はACアダプタ
USB規格:	Ver,2.0対応
対応OS:	Windows98SE/ME/
	Windows2000/XP(SP3)/VISTA/7
	※32ビット、64ビット両方に対応
対応デバイス:	dsPIC30Fシリーズ
	※DIP形状のデバイスの限る
生産国:	セルビア/日本

サポート情報

当方では、dsPICF-400のサポートを行っております。以下のいずれかの方法でご質問をお寄せください。

■FAX番号　03-3700-3548
■電子メール　support@microtechnica.net

なお、他社製品に関することや自作回路に関するご質問にはお答え致しかねますのであらかじめご了承ください。

ソフトウエアはアップグレードされることがあります。アップグレードされると対応デバイスが増えたりバグが修正されたりします。
アップグレードの情報などは弊社のwebページにてお知らせ致しますので、弊社webページを定期的にご確認ください。
http://www.microtechnica.net/

PICマイコン用の統合開発環境,MPLABはマイクロチップ社製のソフトウエアです。バージョンアップ等の情報についてはマイクロチップ社のホームページより告知されますので,下記ページを定期的にご確認ください。
http://www.microchip.com/ (英語サイト)

マイクロテクニカ　microtechnica
〒158-0094　東京都世田谷区玉川1-3-10
TEL: 03-3700-3535　FAX: 03-3700-3548

dsPIC30Fマイコンの概要

dsPIC30Fシリーズは、米マイクロチップテクノロジー社が開発しているPICマイコンシリーズの16ビットマイコンです。マイコンの機能だけでなくデジタル信号処理の機能(DSP)も内蔵しています。

dsPICには、本dsPICF-400にて開発ができるdsPIC30Fシリーズと、さらにメモリー容量や機能を拡張したdsPIC33Fシリーズがあります。その他、DSP機能を搭載しない16ビットマイコンのPIC24シリーズもあります。
※dsPICF33シリーズと、PIC24シリーズに対応した統合開発ボードはdsPICF-900として、当方で販売しております。

dsPICシリーズの詳しい内容については本書ではとても掲載しきれません。専門の書籍やデータシートをご参照ください。本書では、大まかな概要と、dsPICF-400とC言語を使用した簡単なチュートリアルを収録し、いち早く16ビットの世界を体験できるよう内容を編集しました。

■dsPIC30Fシリーズの特徴
dsPIC30Fシリーズは+2.5V～+5.0V動作が可能で、処理速度は、最高で40MIPSと16ビットマイコンではかなりの高速処理を実現しています。また、処理速度を落とすと、低電圧動作も可能となります。命令長は24ビットで、データバスの幅が16ビットになっています。

動作用のクロック源は、従来通り外部発振子も利用できる他、内蔵発振子も使用できます。また内部では、クロックを4倍・8倍・16倍に逓倍する回路も搭載しており、高速な動作が可能となっています。

デジタル信号処理では、外部から入力されたアナログ信号を内部で高速にデジタル処理をするため高速なADコンバーターが要求されます。高精度な12ビットのADコンバーターにより最高で200kspsという高速サンプリングができます。また、10ビットのADコンバーターでは最高で1Mspsの高速サンプリングができ、高い周波数のアナログ信号を扱うことが可能となっています。また、演算処理も高速化しています。ハードウエアにて乗算ができ16ビット同士の演算はクロック1サイクルで完了します。

dsPICシリーズの最大の特徴はMPU機能の他にDSPの機能を内蔵したことです。MPUは16ビットで、MPUがDSPを司るような仕組みになっています。
DSP機能としては、40ビット長のアキュムレータ(演算結果を記憶しておくためのレジスタのことで別名"累算器"ともいいます)を2個、40ビットのバレルシフタ(入力データのビット配置を入れ替えるもの)を1個内蔵しています。これにより積和演算を1サイクルで処理できるようになっています。DSPの構造図は、dsPICのデータシートに記載がありますのでご参照ください。

dsPIC30Fシリーズ並びにdsPIC33Fシリーズは用途に合わせて大きく3つに分かれています。モーター制御用、センサー用、汎用用途と分かれています。それぞれ特徴がありますので使用する際には、データシートをご覧ください。なお、本dsPICF-400に標準で付属しているdsPIC30F4013は汎用シリーズのデバイスで、16ビットのタイマーを5つ、12ビットのADコンバーターを13チャンネル搭載しています。

C言語とは

C言語は、ANSIという団体によって規格化された高級言語です。高級言語とはアセンブラ言語のように機械語に近いレベルの開発言語ではなく、より人間がわかりやすい言語体系の開発言語を指します。C言語を導入することでより簡単に、合理的にPICマイコンの開発ができるようになります。
C言語では様々なデータ型を扱えるほか、データを指し示すポインタや配列、関数などが使用でき、より高度なプログラムを記述することができます。

本dsPICF-400に付属のmikroCコンパイラーはANSI規格に準拠したPICマイコン用のC言語です。本製品に付属のCD-ROMに収録のmikroCはdsPIC30F及びdsPIC33Fシリーズをはじめ、PIC24にも対応しています。体験版のため開発できるプログラムのコードサイズが6KBまでに限定されていること以外、すべての機能が使用できます。mikroCが気に入れば、コード制限のないフル機能版を入手することができます。

mikroCは、一般的なC言語としての機能の他にPICマイコンの機能をフルに使用できるよう様々な組み込み関数を搭載しています。組み込み関数を使用することで、LCD制御やADコンバーターの操作、UARTなど様々な機能を簡単に実現できます。PICマイコンの開発に便利な様々な組み込み関数が搭載されています。

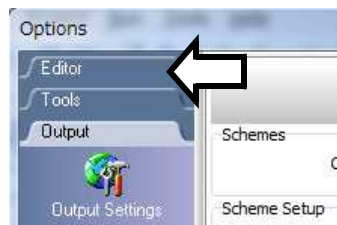
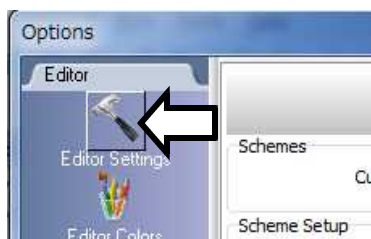
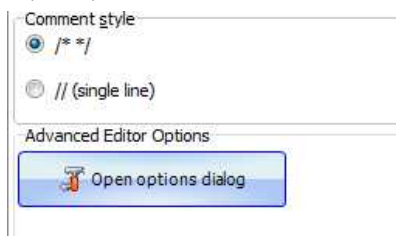
チュートリアル　～dsPICF-400をmikroCで使おう～

早速付属のmikroCを使ってC言語でプログラムを作り、実際にdsPICF-400のボード上で動作させてみましょう。本チュートリアルを一通り試すことで、dsPICF-400の使い方及びmikroCの使い方を体験・習得できます。
なお本マニュアルではC言語の基本から解説することはできませんので、C言語の基本については専門書籍をご参照ください。

■開発環境の設定
では最初にmikroC Cコンパイラーの開発環境を設定しましょう。

1 mikroCを起動します。
"スタート"→"プログラム"→"Mikroelektronika"→"mikroC PRO for dsPIC"→"mikroC_dsPIC"の順でクリックして起動します。

2 起動すると、開発画面が表示されます。
初期設定では、文字が小さく見にくいのでフォントサイズを大きく変更しましょう。

3 メニューバーから"Tools"→"Options"をクリックします。ダイアログが表示されますので、左側のタブから"Editor"をクリックします。

4 続いて、"Editor Settings"のボタンを押します。

5 "Open Option dialog"ボタンをクリックします。

6 "Editor Options"ダイアログが表示されますので、画面右下の"Editor Font"部分にある"Font"ボタンをクリックすると、フォントを選択できるダイアログが表示されますので、サイズを調節してください。

※実用的な大きさとしては、画面の解像度にもよりますが、14ポイント程度が見やすいです。
※右上にある"Font"ボタンではありませんので注意してください。
※フォントの種類は変更しないでください。

設定したら"OK"ボタンを押して完了します。さらにOptionsの画面では、"Apply"ボタンをクリックして設定内容を反映させた後、"OK"ボタンを押して完了します。

※その他にも詳細な設定が可能ですが、トラブルを避けるためフォントの変更以外は行わないことをお奨めします。

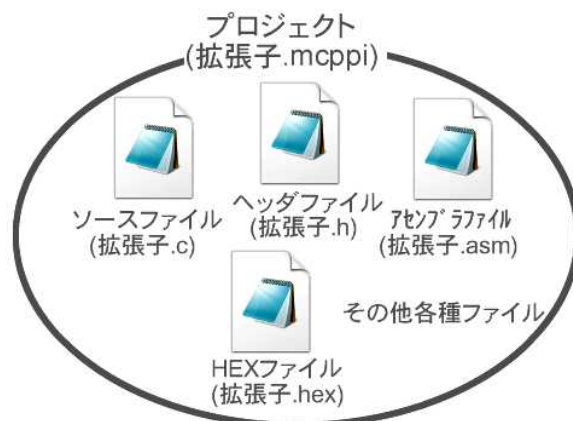
チュートリアル① ～PORTBの制御とボタン処理～

では早速プログラムを作って開発を体験してみましょう。最初のチュートリアルでは、まず全体の体験ということで、簡単なプログラムを作ります。

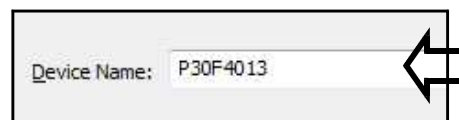
16ビットの変数を用意して、そこに入れた値を2進数でPORTBに出力するプログラムを作ってみましょう。変数を2つ用意して、外部のスイッチ入力によって、どちらの変数の値をRBに出力するのか指定できるようにしてみます。

デバイスは標準で付属しているdsPIC30F4013を使用することにしましょう。

【1】プロジェクトの新規作成

プログラムを作り始める前に、プロジェクトを新規に作成します。mikroCでは1つのファームウエアに対して1つのプロジェクトとして、プロジェクト単位で管理します。
プロジェクトには、これから作成するC言語のソースプログラム(拡張子.c)や、場合によって使用することとなるヘッダーファイル(拡張子.h)などのファイルが登録されます。その他、コンパイル後のHEXファイルなどもプロジェクトに含まれる1ファイルとして管理されます。

よくある間違いとして、プロジェクトの意味を理解していないと画面上に表示されているソースプログラムをコンパイルしているつもりなのに、実際には別のソースプログラムがコンパイルされていて思ったような動作をしなかったり混乱してしまったりというトラブルに繋がります。新しいプログラムを作るときには、必ずプロジェクトを新規に作成するようにしてください。

1 新規にプロジェクトを作成してプログラムを作っていきましょう。
メニューバーから"Project"→"New Project"をクリックします。
mikroCではウィザード形式でプロジェクトを新規に作成します。ウィザード画面が表示されたら、"Next"ボタンを押して続行します。

今回使用する"P30F4013"を選択して、"Next"ボタンを押します。

2 続いてクロック周波数を設定します。今回は、dsPICF-400に最初から付いている10MHzを使用することにします。"Device Clock"の部分に、"10" と入力して 10MHz に設定します。
小数点以下は自動的に0が挿入されますので、単に 10 とだけ入力してください。
"Next"ボタンを押します。

3 プロジェクトの保存場所と、プロジェクト名を指定します。
ダイアログの右端にあるファイルボタンを押すとディレクトリを指定するダイアログが表示されます。プロジェクトを作成するディレクトリと、プロジェクト名を入力して"保存"ボタンを押します。

 ディレクトリ名及びファイル名には、日本語や全角文字などの2バイト文字は使用できません。必ず半角英数字で指定てきるディレクトリ及びファイル名としてください。

ここでは、例としてCドライブの直下に"mikroC_project"というフォルダを作り、そこに"TEST01.mcpds"というプロジェクトを作成することにします。ディレクトリ名は次のようになります。
C:¥mikroC_project¥TEST01.mcpds

4　続いて "Step4/6"、"Step5/6" のウィザードでは何も変更等はせずに"Next"ボタンを押します。
最後に"Finish"ボタンをクリックして完了します。

【2】プログラムの記述

ではプロジェクトが作られましたので、プログラムを記述しましょう。mikroCでは、新規にプロジェクトを作成すると、"プロジェクト名.c"という拡張子がcのファイルが自動的に作られます。基本的にはこのファイルにC言語のプログラムを記述していくことになります。

新規にcファイルが作られると、main関数だけが記述された状態になります。では早速プログラムを記述してみましょう。

C言語では大文字・小文字は基本的には関係ありません。但し、統一した方が見やすいので、一般的には関数や変数は小文字で、レジスタ名は大文字で記述することが多いです。

プログラムは見やすいようにタブやスペース、改行を適宜入れながら記述していきます。コンパイラは、スペースや改行は無視します。

```
void main(){

  unsigned int a , b ;
  TRISB = 0 ;
  TRISD = 0x3;
  PORTB = 0 ;
  a = 5200 ;
  b = 30000 ;

  while(1){
     if (PORTD == 0x1) PORTB = a ;
     if (PORTD == 0x2) PORTB = b ;
  }
}
```

【3】コンフィギュレーションビットの設定

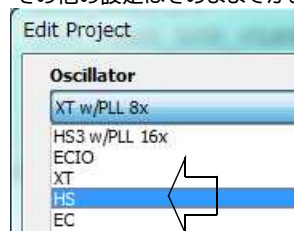
PICには、コンフィギュレーションビットという特殊なレジスタがあります。これは、PICの動作に関わる最も基本的な設定を行うもので、プログラムをPICに書き込む時に設定するものです。

代表的なものには、例えば発振子の種類の指定があります。発振子は大きく分けて外部発振子か、内蔵発振子かに別れ、さらに外部発振子の場合にはその周波数によって、設定が分かれます。その他、ウォッチドッグタイマーのON/OFFの設定などを行うのがコンフィギュレーションビットです。

mikroCでは、コンフィギュレーションビットの設定は、生成するHEXファイルに埋め込んだ形でコンパイルを行います。HEXファイルにコンフィギュレーションビットの設定を埋め込んでおけば、書き込みの時、いちいち設定をしなくても、書き込みソフトウエアはHEXファイルを読み込むと自動的にコンフィギュレーションビットの値をセットするので、手間が省け、間違いも少なくなります。次の手順でコンフィギュレーションビットの値を設定しましょう。

1　メニューバーから"Project"→"Edit Project"を選択します。

2　ダイアログが表示されます。細かな設定は色々とありますが、すべて解説することはできませんので、ここでは必要最低限の設定のみ設定します。最初に最も基本的な設定を読み込みますので、"Default"ボタンをクリックします。

3　プルダウンの一番上"Oscillator"の所をクリックして、プルダウンから"HS"を選択します。
その他の設定はそのままでかまいません。

このOscillatorとは、PICの発振子の種類を指定するもので、外部発振子の場合、水晶発振子で4MHzの場合にはXTに、5MHz以上又は発振子にセラミック発振子を使用している場合には、HSに設定します。dsPICF-400のボードには、標準で10MHzの水晶発振子が取り付けられていますので、ここでは、HSに設定をしたというわけです。その他諸処設定はありますが、ここではデフォルト設定を使用します。

※コンフィギュレーションビットの設定については、デバイス毎に異なります。データシートに詳しい記載がありますので、使用するデバイスのデータシートを必ず参照ください。コンフィギュレーションビットの値が間違っていると、プログラムが正しくても動作しなかったり、予期しない動作をしたりすることがあります。

4　設定が完了したら、"OK"ボタンを押して完了します。
なお、以後すべてのチュートリアルで本設定をする必要がありますので、この設定について覚えておいてください。

【4】ボードの設定とコンパイル及び書き込みの実行

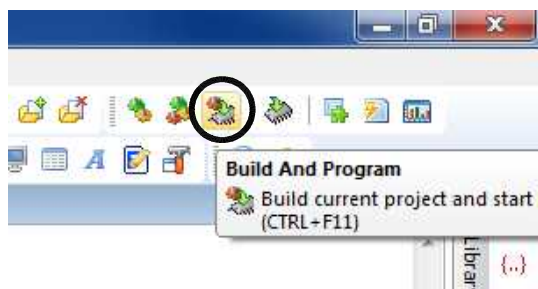
C言語で書かれたプログラム(ソースプログラムといいます)を、PICに書き込める機械語のHEXファイルにする作業をコンパイルと呼びます。mikroCとdsPICF-400の組み合わせの場合、コンパイルとコンパイル後に生成されたHEXファイルの書き込みを連動して行うことができます。コンパイルでエラーが発生した場合には、書き込みは実行されません。

ここでは最初にボードの設定を行った後、コンパイル→書き込みを実行してみましょう。書き込み後はすぐにプログラムが動きますので、書き込みを実行する前に、dsPICF-400のボードの設定をしておく必要があります。

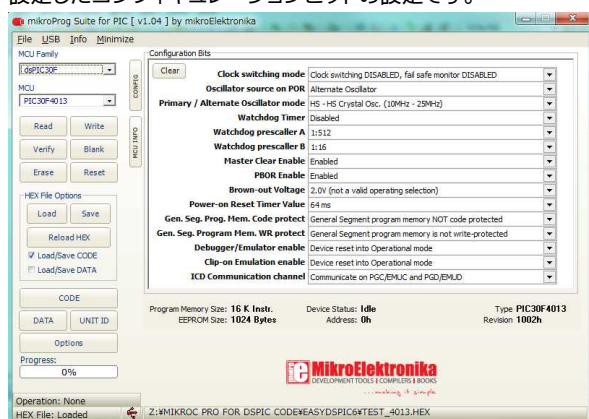
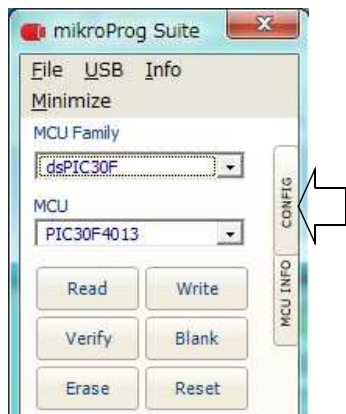
1　dsPICF-400ボードの各種設定を行いましょう。今回はRBはLEDに接続し、RDは下位ビットを定常時プルダウンに設定し、タクトスイッチが押された時にVcc(5V)に接続されるようアクティブHigh設定にします。次のように設定してください。

・ディップスイッチSW12
　→"PORTB&C"(1番)のみスイッチをON側にします。
　　その他のスイッチはすべてOFF側にセットします。
・ジャンパーJ3(PORTD&A)
　→"Pull-Down"間(下側)をジャンパーソケットでショートします。
・ディップスイッチSW3
　→すべてON側にします。
・ジャンパーJ15(タクトスイッチ設定)
　→"VCC"側をジャンパーソケットでショートします。

上記のように設定すると、PICのPORTBのピンはLEDに接続され、該当のビットがHレベルになるとLEDが点灯します。
また、PORTDは常時Lレベルで、タクトスイッチが押されると、該当ビットがHレベルになります。

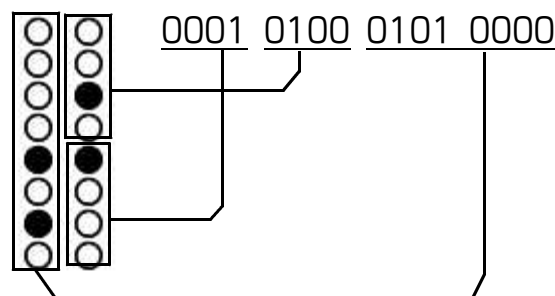
2　設定ができたら、コンパイルと書き込みボタンをクリックします。

ツールバーの"Build and Program"ボタンを1回クリックします。

3　クリックするとコンパイルが始まります。
ウインドウ下部のメッセージウインドウにコンパイル結果が表示されます。エラーがなければ、"Finished successfully"と緑の文字で表示されます。
エラーが表示された場合には、プログラム内容に間違いがないか確認してください。
生成されたHEXファイルは、プロジェクトを保存したフォルダ内に生成されています。

4　コンパイルが正常に完了すると自動的に書き込みソフトウエアが起動し、書き込みを実行します。進捗状況がプログレスバーで右下に表示されます。
"Configuration Bits"と書かれたプルダウンのまとまり部分が先に設定したコンフィギュレーションビットの設定です。

※mikro Prog Suiteの画面表示が小さい(コンフィギュレーションビットの設定ウインドウが表示されない)場合には、mikro Prog Suiteの"Config"ボタンをクリックすると、画面が展開します。

【5】動作確認

書き込みが完了するとプログラムがdsPICF-400ボード上ですぐに動きます。動作を確認してみましょう。

dsPICF-400のタクトスイッチ群のなかにあるRD0のスイッチを押します。RBに接続されたLEDが点灯します。RE0なので変数aの値、ここでは5200が2進数でLEDに現れます。

5200は2進数表記では　0001010001010000　となります。
左側が上位ビット、右側が下位ビットです。RB15が上位ビット、RB0が下位ビットになりますので、下記のように点灯しているはずです。

※○は消灯、●は点灯

同様にして、RD1のスイッチを押すと変数bの値、すなわち30000が2進数としてRBのLEDに表示されます。

変数の値を色々と変えたり、スイッチの種類を増やしたりして実験を行ってみましょう。思い通りにプログラムが動作することを確認してください。

【6】プログラムの解説

void main(){…}

main関数です。C言語では1つのプログラム内に必ず1つのmain関数が必要です。voidはmain関数の型で戻り値がない関数であることを宣言しています。{ }で囲まれた範囲がmain関数の範囲となります。

unsigned int a , b ;

符号なしのint型変数を宣言しています。C言語では文の最後には必ず ; (セミコロン)をつけます。
符号なしのint型の場合扱える値は、0～65535までです。

TRISB = 0 ;
TRISD = 0x3;

TRISレジスタは、ピンの入出力方向を設定するレジスタです。
該当のビットが0の場合は出力、1の場合は入力となります。本チュートリアルではRBは全ピン出力、RDはRD0とRD1が入力なので0x3として設定しています。
C言語では16進数の値を表現する時は数値の前に0xを付けます。

PORTB = 0 ;

PORTBに0を代入しています。=(イコール1つ)は代入演算子で右辺の値を左辺に代入します。mikroCでは「レジスタ名=値」の書式でレジスタに値を代入できます。

a = 5200 ;
b = 30000 ;

int型変数に値を代入しています。
代入は、右式から左式に代入されます。代入には代入演算子(=)を用います。

while(1){…}

while文は(　)内の値が1又は真であれば{ }内の処理を繰り返す繰り返し文です。ここでは評価式を1としていますので永久に{ }内が実行されます。

C言語では処理を永久ループさせたい場合にはよくwhile文を使用します。

if (PORTD == 0x1) PORTB = a ;

if文を使用した入力判定部分です。

()内の評価式が真の時にその後ろに記述された文が実行されます。mikroCでは、レジスタの値を直接評価式の中で判定することができます。

==(イコール2つ)は比較演算子の「等しい時」を表す演算子です。

よくある間違いとして=を1つしか記述しない場合があります。=が1つの場合には代入演算子となってしまい正しく動作しません。文法上は間違いではないのでエラーがでないため間違いを見逃してしまいプログラムが思うように動かない・・・ということがあります。

PORTDは、スイッチが押されるとそのビットがHになります。RD0のタクトスイッチが押されると、0000 0000 0000 0001 となり、PORTDの値は1となります。同様にして、RD1のタクトスイッチが押されると、0000 0000 0000 0010 となりますので、PORTDの値は2となります。これをif文で判定して処理を分岐しているわけです。

PORTB = aで、変数aの値をそのままPORTBレジスタに代入しています。

チュートリアル②　～LCDの使用と、配列・ポインタ～

■プログラムの作成

dsPICF-400に装着されている2行16文字のLCDに文字列を表示します。C言語では、文字列を扱う場合配列を使用します。

固定した文字列をLCDに表示させてみましょう。

プログラムの作り方は、チュートリアル①と同じです。新規にプロジェクトを作成し、新しいプログラムとして開発します。"Edit Project"からコンフィギュレーションビットの値を設定するのを忘れないようにしてください。

```
sbit LCD_RS at LATD0_bit;
sbit LCD_EN at LATD1_bit;
sbit LCD_D4 at LATB0_bit;
sbit LCD_D5 at LATB1_bit;
sbit LCD_D6 at LATB2_bit;
sbit LCD_D7 at LATB3_bit;

sbit LCD_RS_Direction at TRISD0_bit;
sbit LCD_EN_Direction at TRISD1_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB3_bit;

void main(){
     char txt1[] = "HELLO";
     char txt2[] = "WORLD";

     ADPCFG = 0xFFFF;

     Lcd_Init();

     Lcd_Cmd(_LCD_CLEAR);
     Lcd_Cmd(_LCD_CURSOR_OFF);
     Lcd_Out(1,1,txt1);
     Lcd_Out(2,1,txt2);
}
```

プログラムを記述した後はチュートリアル①と同様の手順でコンパイルをして、HEXファイルを生成後、書き込みを実行します。ボードの設定は特に注意する点はありませんが、ディップスイッチはすべてOFFにしておいたほうがよいでしょう。

LCDの1行目に"HELLO"と表示され、2行目に"WORLD"と表示されれば成功です。

※ディップスイッチとジャンパーソケット

dsPICF-400にはdsPICマイコンと各種周辺回路を物理的に接続するかどうかを設定する多数のディップスイッチとジャンパーソケットがあります。効率よく、かつ他の回路への影響を最低限にしながら各種実験ができるようになっているためです。

しかし場合によっては、思いがけずスイッチがONになっていたり、ジャンパーソケットによってジャンパーされていると、そちらの回路側に電流が回り込み、流れる電流が多いと電圧降下が発生して、思い通りの動作が得られない場合があります。使用しない回路からは、必ず切り離しておくようにすると、思いがけないトラブルを避けることができます。

■プログラムの解説

sbit LCD_RS at LATD0_bit; ～
sbit LCD_D7_Direction at TRISB3_bit;
main関数の前にsbitタイプを用いてLCDの物理的な配線状態を定義します。
sbitタイプは、PICマイコンのSFR(レジスター)の特定のビットに対してアクセスするため機能を提供します。
ここでは、その1つの使い方としてLCDの接続状態をあらかじめ定義しています。sbitは、TRISレジスタに対しても特別な書式として記述することができます。LCD関連のライブラリを使用する場合には、main関数の前で、この定義を記述してください。
この配線記述は、dsPICF-400のボード配線にあわせて記述しています。

※この定義をmain関数の中で記述してしまうとエラーになります。

char txt1[] = "HELLO";
char型配列を宣言しています。char型は文字を扱う8ビットの型です。C言語では文字はASCIIコードにて扱われます。
通常配列を宣言する際には要素数(インデックス値)を[ ]内に入力しますが、[ ]内を空欄にすると自動的に要素数が設定されます。配列や変数を宣言する際に初期値を代入できますが、char型に限り文字列で初期化できます。"(ダブルクォーテーション)で囲むと文字列として認識されます。

ADPCFG = 0xFFFF;
ADPCFGレジスタは、16個あるADコンバータ入力ピンのうち、どのピンをデジタルI/Oとして使うか、アナログ電圧入力として使うかを設定するレジスタです。アナログ入力にするピンに対応するビットは0にします。ここではすべてのピンをデジタルI/Oとして使用しますので、0xFFFFを代入しています。

Lcd_Init() ;
LCDを初期化する関数です。LCDを使用する前に記述します。

Lcd_Cmd(_LCD_CLEAR);
Lcd_Cmd( )関数は、LCDの制御コマンドを送信する関数です。
ここでは、画面をクリアーするコマンドと、カーソル(描画位置に点滅するカーソル)を表示しない設定にしています。
※mikroCのマニュアルは、"Help"→"Help"で見ることができます。

Lcd_Out(1,1,txt1) ;
Lcd_Outは4ビットモードでLCDに文字や文字列を表示する時に使用します。書式は下記の通りです。

Lcd_(行, 列, char *text);

*textは、char型のポインタという意味です。ここでは、配列名を記述することで配列の0番目の要素のアドレス値を指定しています。

*textは、char型のポインタという意味です。ここでは、配列名を記述することで配列の0番目の要素のアドレス値を指定しています。
なお、ポインタを使用して次のページのようにプログラミングしても同様の動作をします。ポインタについての詳細はC言語の専門書籍を参照ください。

チュートリアル③ ～非同期式シリアル通信とADコンバーター～

■プログラムの作成

非同期式シリアル通信(UART)を使用して、RS232C通信をするプログラムを作ります。搭載のdsPIC30F4013は、TXがRF3、RXがRF2にアサインされています。

RS232C経由で文字'A'(0x41)を受信すると、ADコンバーターのチャンネル4(AN4)の値を8ビットの値で出力します。同様にして、文字'B'(0x42)を受信すると、文字列"HELLO"を出力します。

なおUARTの通信速度は9600bps、データ長8ビット、1ストップビット、ノンパリティの設定とします。

```
void main() {

  char uart_rd;
  unsigned char AD_A;

  TRISB = 0xFFFF;
  UART1_Init(9600);
  Delay_ms(100);

  while (1) {
    AD_A = ADC1_Read(4);

    if (UART1_Data_Ready()) {
      uart_rd = UART1_Read();
      switch(uart_rd){
       case 'A': UART1_Write(AD_A);break;
       case 'B': UART1_Write_Text("HELLO");break;
      }
    }
  }
}
```

■dsPICF-400ボードの設定

このプログラムを正しく動作させるため、下記のようにdsPICF-400ボードの各種設定をセットしてください。ボードの設定が正しくないとプログラムは動作しませんので、設定内容とプログラムを比べて確認してください。設定としてはRS232Cを使用できるようにすることと、ADコンバータのAN4を使用できるようにすることです。

・すべてのディップスイッチはすべてOFF側に設定します。
・SW7はRXの"RF2"(2番)、TXの"RF3"(6番)のスイッチをONにします。
・LCDはPORTBと接続されているため取り外してください。
・J12は"RB4"をジャンパーソケットでショートします。

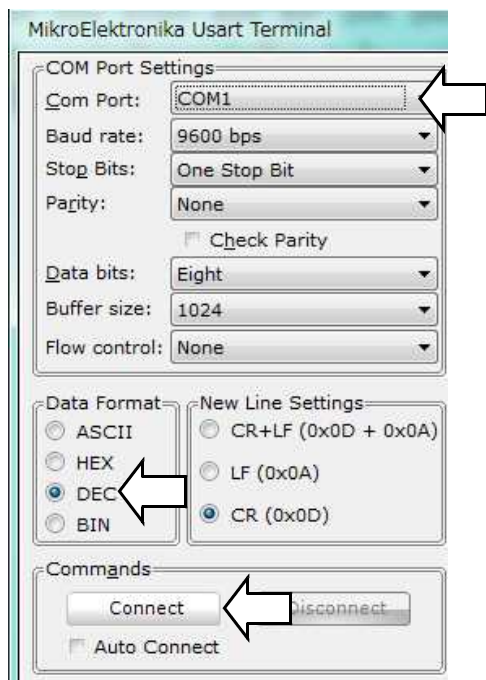
■コンパイルと書き込みの実行

これまでのチュートリアルと同様、ソースプログラムをビルドして、HEXファイルをdsPICF-400のマイコンに書き込んでください。

■パソコン側の準備

このプログラムでは、RS232C通信を行います。パソコンのRS232Cポートと、dsPICF-400のRS232Cポートを全結線ストレートケーブルで接続してください。

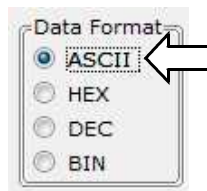
RS232CのターミナルソフトはmikroCに付属しています。このターミナルソフトを使用すると大変便利に通信が行えます。
mikroCのツールバーにある"USART Terminal"のアイコンをクリックします。

ターミナルソフトが起動します。
"Settings"グループにて通信に関する基本的な設定を行います。
今回は、通信速度9600bps、1ストップビット、ノンパリティーの設定にしますので、下図のように設定してください。
"Com Port"は、dsPICF-400と接続したRS232Cのポート番号を設定してください。通信速度(Baud rate)が9600bpsになっていることを確認します。
また、最初の実験では文字Aを送信して、戻り値としてADコンバータの値を10進数で表示しますので、その下の"Data Format"のチェック部分について、"DEC"にチェックを入れてください。

設定が終わったら、"Connect"ボタンをクリックします。

■動作確認

ターミナルソフトの"Connect"ボタンを押すとRS232C通信が開始されます。最初にADコンバータで取得した値を10進数で取得してみましょう。
ターミナルソフトの"Send"のテキストボックスに大文字で　A　と入力して"Send"ボタンをクリックします。

すると、その下の"Receive"ウインドウに値が表示されます。試しにP1ボリュームを回転させて、'A'を何回か送信してみましょう。回転させる毎に異なった値が返ることが確認できます。ボリュームを左に回しきった状態の時は　0　が返ります。右いっぱいに回しきると255が返ります。

同様にして、'B'も試してみましょう。文字Bを送信すると、文字列"HELLO"ほ返します。これは文字列ですので、受信するデータフォーマットは、ASCIIコードですので、ターミナルソフトの"Data Format"の部分の所で、"ASCII"をチェックしてください。

この設定で、'B'を送信すると、Receiveウインドウに"HELLO"と表示されます。dsPICは、正しく受信したデータによって返す値を分岐して処理していることが確認できました。

なお、dsPICのADコンバーターの分解能は10ビットですが、シリアル通信ではデータ長がは8ビット長なので、最大値は255となり、255を過ぎると、0に戻ります。ボリュームのちょうど中間部分が0になります。右いっぱいに回すと255になり、ちょうどボリュームが中間位置だと0が返ることが分かります。

■プログラムの解説

TRISB = 0xFFFF;
PORTBは全ビット入力ピンに設定しています。今回は、AN8とAN12だけ入力ならばよいので、TRISB=0x1100 としてもよいです。

Uart1_Init(9600);
UART1の初期化を行う関数です。()内には通信速度をbps単位で記述します。ここでは9600bpsに設定しています。

delay_ms(100);
UARTのユニットが初期化するのに約100ミリ秒かかりますので、100ミリ秒の遅延を挿入しています。delay_ms( )関数は、引数で指定した秒数(単位はミリ秒)だけ遅延を作ります。

AD_A = ADC1_Read(1);
Adc1_Read()関数は、( )内に指定したチャンネルからAD変換の値を取得して返します。本来dsPICシリーズのADコンバーターは分解能が10ビット長ですが、ここでは、この値を後にUARTで送信する関係上、符号無しの8ビット長変数(char型)に代入しています。
ここでは8を指定してるため、AN1(RB1)に印加された電圧値の値を取得します。

if (Uart1_Data_Ready()) {
Uart1_Data_Ready()関数はUARTの受信バッファにデータが格納されると真になります。これをif文で検出することでデータの受信を監視します。

uart_rd = Uart1_Read()
char型の変数uart_rdに受信データを取り込みます。1バイトサイズのデータのみ受信できます。連続するデータの場合には、配列とポインタを使用します。

switch (uart_rd){
switch文は、( )内の評価式を評価してcaseで指定された値のルーチンに処理を分岐します。評価式にuart_rd変数を指定し、UARTで受信したデータの内容によって処理を分岐しています。
caseの後には値を記述し、評価式がその値になると、そこにジャンプします。ここでは、'A'及び'B'を指定することで、受信した文字によって処理を分岐しています。
C言語では、1文字の場合には ' (シングルクォーテーション)で囲います。(文字列はダブルクォーテーションです。)
なお、break文を記述すると、switch文から処理が抜け出します。

Uart1_Write(AD_A);
Uart1_Write()関数は、UART経由で( )内のデータを出力します。( )内は符号なしのshort型又はchar型変数が指定できます。
ここでは、ADC1_Read()関数で取得した値をUARTで出力しています。

UART1_Write_Text("HELLO")
Uart1_Write_Text()関数は、( )内の文字列をUARTで出力する関数です。文字列は、ダブルクォーテーション(")で囲みます。

チュートリアル④　～TMR1を使用した割込プログラム～

■TMR1とは‥
dsPIC30Fシリーズのデバイスには、TMR1という内蔵タイマーが搭載されています。TMR1は、16ビットのカウンターで、内部クロックか外部クロックに同期させてカウントを行わせることができます。
外部クロックの場合には、32.768KHz用の発振回路が内蔵されておりリアルタイムクロックとして使用できる仕様となっています。

TMR1は、PR1レジスタの値と常に比較されるようになっており、TMR1の値と、PR1レジスタの値が一致した時、割込が発生する仕組みになっています。また割込発生後は、TMR1の値は0に初期化されます。

■dsPIC30Fの割込について
dsPICでは、割込の種類(何のイベントで発生した割込か)によって、ジャンプするベクタ(ジャンプベクタと呼びます)が異なるベクタ方式という割込を採用しています。8ビットのPICマイコンに比べ格段に構造が複雑になっています。これにより、様々な割込に応じて異なる割込ルーチンを作成でき、より複雑な割込プログラムを作ることができます。割込には、62個の割込が用意されています。

ジャンプベクタは、プログラムメモリーのアドレスに割り当てられています。割込が発生すると、その割込の種類によって、ジャンプベクタが変わります。どんな割込が発生した時、どのジャンプベクタへジャンプするのかは決められており、例えば今回使用するTMR1割込の場合には、ベクタ番号は11となります。このように割込の種類と、飛び先アドレスの一覧を主ベクタテーブル(IVT)と呼びます。IVTは、プログラムメモリーの0x0004～0x007Eに配置されています。

IVTの他にもう1つベクタテーブルが用意されており、これを副ベクタテーブル(AIVT)と呼びます。これは、プログラムメモリーの0x0084～0x00FEに配置されています。これは、IVTが何らかの事情で使用できない場合(※1)に、IVTに変わってAIVTを使用することができることを意味します。基本的には、IVTを使用すると覚えておいてください。

※1:dsPICでは、プログラムメモリーにデータを書き込む場合、96バイト単位で書き込まなければならないという決まりがあります。
ITVに書き込みを行おうとすると、同時にリセットベクタも書き込みを行う必要が出てきます。例えばブートローダーを書き込んでいる場合などでは、リセットベクタ(0x0002番地)とIVTが隣り合っているため、IVTを書き換えるには、リセットベクタも同時に書き換えを行わなくてはならなくなり、ブートローダーで不具合が発生した場合、リセットベクタが不正になってしまうことがあるため、IVTにはさわらずに、割込を使用する場合には代わりにAIVTを使用することがあります。

また、dsPICになってから割込に順位が付くようになりました。これまでの8ビットのPICでは、割込は同時には1種類だけしか選択できませんでしたから、発生した割込の順位などは最初から必要ありませんでした。しかしdsPICでは複数の割込をベクタ方式で処理できるため、割込に優先順位を持たせてあります。
優先順位は、レベル0～レベル15までの16段階です。15が最も優先順位が高い割込になります。TMR1割込のような汎用割込の場合には、レベル7～レベル0が設定されます。

では順位はどのように決まっているかというと、IVTの並び順となっています。IVTの値が小さい方が、レベルが高い、すなわち優先的に割込として受け付けられるということです。
例えば、TMR1の割込はIVTでベクタ番号11です。対してADコンバーターの読み取り完了時に発生する割込はベクタ番号19です。よって、2つの割込が同時に発生した場合には、TMR1の方が優先されるということになります。

さらにCPUには割込許可レベルというものがあり、これも0～15で設定します。これは、CORCONレジスタのビット3と、SRレジスタのビット2～0の計4ビット(CORCONレジスタのビット3が最上位ビット)で指定するもので、ここで設定した値よりも、優先順位が高い割込が、実際の割込として許可される仕組みになっています。

■プログラムの内容
TMR1割込を使用して、200ミリ秒毎に割込を発生させて、PORTBの出力を、200ミリ秒ごとに反転させるプログラムを作ります。

■プログラムの仕組み
今回は、dsPICF-400のボードに最初から装着されている10MHzの水晶発振子をクロックとして使用します。クロックは逓倍せずにそのままCPUのクロックとして10MHzとして使用します。(コンフィギュレーションビットの設定では、オシレーターをHSに設定します)

TMR1のクロック源は内部クロックとします。dsPIC内部では動作クロックが1/4になります。よって、内部は2.5MHzで動作しています。
時間に直すと1周期400nsです。タイマーの時間は下記の式で求められます。

TMR1時間=(1/4×動作クロック) × プリスケーラー分周比 ×
(PR1レジスタの設定値-1)

今回は下記の式により約200ミリ秒を作ります。

400ns×256×(1954-1) = 199.9872ミリ秒

きれいに割り切れないため、若干の誤差が生じますがこれは、クロック周波数を見直すことできれいに割り切れる値になります。
今回は、10MHzのクロックを使用している関係上、若干の誤差が生じる計算になります。

■プログラムの作成
新しいプロジェクトを作成して、プログラムを作ってください。プロジェクトの作成方法等は従来の方法と全く違いはありません。

```
void Timer1Int() iv IVT_ADDR_T1INTERRUPT{
  T1IF_bit = 0;
  LATB = ~ PORTB;
}

void main() {
  ADPCFG = 0xFFFF;
  TRISB = 0;
  LATB = 0;

  T1IF_bit = 0;
  T1IE_bit = 1;
  T1CON = 0x8030;
  IPC0 = 0x7000;
  PR1 = 1954;
}
```

■dsPICF-400ボードの設定
SW12の"PORTB&C"のスイッチだけをONにしてその他はすべてOFFに設定します。LCDは外しておいてください。

■動作確認
プログラムを書き込むと、PORTBに接続されたLEDが全部200ミリ秒間隔で点滅します。オシロスコープをお持ちであれば、PORTBのいずれかのピンの波形を見ると、200ミリ秒間隔の矩形波を観察できます。

■プログラムの解説

void Timer1Int() iv IVT_ADDR_T1INTERRUPT{
TMR1の割込が発生した時に実行されるルーチンです。
mikroCでは、ivキーワードがあり、ivキーワードに続いて、割り込みの種類を指定することで、その割り込みが発生した場合に、{ }内に記述されたルーチンが実行されます。

通常は、main関数が実行されていますが、割り込みが発生した時にこの関数が実行されます。

T1IF_bit = 0;
TMR1割込発生フラグを初期化しています。IFS0レジスタのビット3に0を代入しています。割込先ルーチンでは、必ず対応する割込発生フラグを初期化しないと、次の割込を発生させられませんのでここで初期化しています。
mikroCでは、レジスタの特定のビットを指定する場合には、
"ビット名_bit"で指定することができ、代入演算子で値を代入できます。

LATB =~ PORTB;
ポートの状態を反転させています。LATBレジスタは、現在のポートの状態ではなく、PORTBレジスタの値をそのまま反映しています。よって、PORTBの状態を毎回反転させることで点滅させています。

T1IE_bit = 1;
IEC0レジスタのビット3にある、TMR1割込の許可制御ビットを1にしてTMR1の割込を有効にしています。

T1CON = 0x8030;
T1CONレジスタは、TMR1の各種設定をするためのレジスタです。
ここでは、「外部ゲートOFF・プリスケーラー分周比256・クロック源は内部クロック」という設定にしています。

IPC0 = 0x7000;
IPC0レジスタでは、割込の優先順位を制御します。
ビット14〜11がTMR1の割込レベルの設定ビットになっており、000で禁止、111でレベル7(最高)となります。
今回は、TMR1割込しか使いませんので、最高レベルの111に設定しています。

PR1 = 1954;
PR1レジスタの値です。TMR1は常にここの値と比較され、ここの値と一致した時に割込を発生させます。
今回は前のページに記載した計算によって求めた値を代入しています。

※今回のプログラムでは、main関数内ではレジスタの設定をしているだけです。main関数内に別の処理を記述していくと、割込が発生するごとに割込ルーチンが実行され、割込が完了すると、main関数が再度実行され、これを永久に繰り返すことになります。

チュートリアル④-2　～圧電ブザーを鳴らしてみよう～

せっかく先のチュートリアルで、PORTBの点滅が実現できましたので、ここではPORTBに圧電ブザーを接続して、ブザーを鳴らしてみましょう。
圧電ブザーは発振回路を内蔵していないため、人間の可聴域の周波数の電圧信号を外部から印加することで音として認識できます。ここでは、1KHzの矩形波をdsPICで作り、圧電ブザーに印加します。プログラムは先のプログラムをそのまま流用します。1KHzということは、0.5ミリ秒毎に割込を発生させればよいことになります。
今回はプリスケーラの分周比を1:64に設定してすこし動作を速くします。そして次式で計算すると次のようになります。

500u = 400n × 64 × (x-1)
x = 20.53・・・

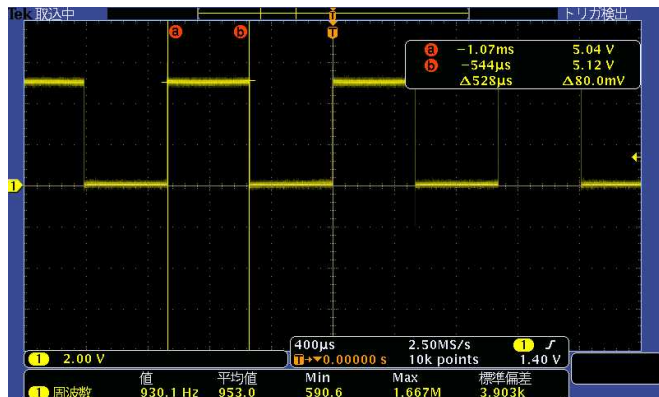
xは20.53・・となりますので、PR1の値は近似値として20にします。きれいに割り切れないたけ、ぴったり1KHzにはなりませんが、約1KHzは作れます。

先のチュートリアルのプログラムの"T1CON = 0x8030;"の部分の値を0x8020に、PR1の値を20としてビルドして書き込んでみましょう。T1CONレジスタは、プリスケーラ分周比を設定できるレジスタですが、ここでは1:64に設定しています。

■dsPICF-400ボードの設定

LEDは使用しませんのでSW12はすべてOFFにしておきます。RB12を圧電ブザーに接続しますので、SW9の3番スイッチだけONにします。

■動作確認

プログラムを書き込むと、圧電ブザーから音が鳴ります。この音は、周波数約1KHzの音になります。オシロスコープをお持ちならば波形を確認してみてください。1周期が約1ミリ秒の波形が観察できます。

チュートリアル⑤　～ICD機能を使ったデバッグを体験する～

■ICD機能とは?

ICDとはインサーキットデバッグの略で、ターゲットボード上のデバイスに実際にプログラムを書き込み、動作させながらパソコン上でデバッグを行う手法のことです。

例えば現在実行している行をハイライト表示させながら実機での動作が見られますので、プログラムの挙動を簡単に把握できます。また、1行ずつプログラムを実行したり、変数やレジスタの現在の値をリアルタイムに閲覧することができたりします。

本セットに付属のmikroCには、C言語レベルでデバッグのできるICD機能が搭載されています。dsPICF-400と組み合わせることで、ICD機能を簡単に体験することができます。

■ICD機能を使うには

ICD機能を使用する場合、プログラムには特に大きな変更点はありませんが、delay_(ms)などのdelay関数が入ったプログラムは正しく動作しません。ICDを使用する場合には、delay関数をプログラム中からなくして実行してください。また、関数によっては、1つの関数で大量の処理を実行するため、ステップ実行(1行ずつ実行)では、時間がかかりすぎる場合があります。

ICD機能を使用する場合、コンパイル時に"Project Settings"から、"Build type"を"ICD debug"設定にする必要があります。このチュートリアルを通してICDの使い方を一通り体験してみましょう。

■プログラムの内容

LCDに文字"microtechnica"を表示されます。但しICD機能の動作を確認するため1文字ずつ表示するプログラムを作ります。

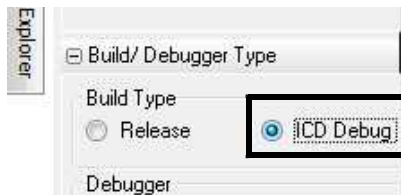
■プログラムの作成

新しいプロジェクトを作成して、プログラムを作ってください。
プロジェクトの作成方法等は従来の方法と全く違いはありません。

```
sbit LCD_RS at LATD0_bit;
sbit LCD_EN at LATD1_bit;
sbit LCD_D4 at LATB0_bit;
sbit LCD_D5 at LATB1_bit;
sbit LCD_D6 at LATB2_bit;
sbit LCD_D7 at LATB3_bit;

sbit LCD_RS_Direction at TRISD0_bit;
sbit LCD_EN_Direction at TRISD1_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB3_bit;

void main(){

   char text[14]="microtechnica";
   int i;

   ADPCFG = 0xFFFF;
   Lcd_Init();
   Lcd_Cmd(_LCD_CLEAR);
   Lcd_Cmd(_LCD_CURSOR_OFF);

   for(i=0 ; i<13 ; i++){
      Lcd_Chr_Cp(text[i]);
   }
}
```

■コンパイルと書き込み

プログラムが書き終わったら、コンパイルをしますが、この時従来とは違い、Build typeをICD用に変更します。画面左下の"Project Settings"タブをクリックします。ウインドウが展開します。

ウインドウが表示されたら、その中の"Build/Debugger Type"の部分にある"Build Type"のチェックボックスのうち、"ICD debug"にチェックを入れます。

チェックを入れたら、従来通り、メニューバーのビルドして、コンパイルが終了したら従来どおり、PICマイコンへ書き込みを行ってください。この作業はこれまでと同じです。

"ICD debug"オプションを指定してコンパイルしたプログラムは、PICマイコンへ書き込んだ後にすぐには動作しません。

■動作の前にボードの確認をしましょう

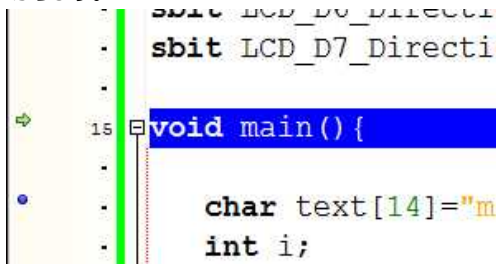
実際にプログラムを走らせる前には、必ずボードの設定状態を確認してください。ボードの設定が正しくないと動作しないことがあります。

・LCDを取り付けてください

■ICDを動作させてみる

ICDを実行して、プログラムの動作内容を確認しましょう。
LCDに何も文字列が表示が出ていないことを確認します。文字列が表示されている場合には一度dsPICF-400の電源を切断してLCDの画面をクリアした状態で下記を実行してください。

1 メニューバーの"Run"→"Start Debugger"をクリックします。
キーボードの"F9"キーでも同様の操作ができます。

2 void main() { の行が青色バーでハイライトされます。
ICDでは実行される行が青色バーでハイライト表示されます。
また、行数表示の左隣に➡マークが表示され、実行している行が分かります。

※void main() { の行が青くハイライトされず、別のアセンブラの画面が表示されてしまった場合には、メニューバーの"Run"→"DisAssembly mode"をクリックしてください。

3 ICDでは、1行ずつ実行する場合、LCD関連のライブラリーは動作が複雑で時間がかかるため、ここでは一気にfor文の所までは通常通りプログラムを動作させてしまいましょう。

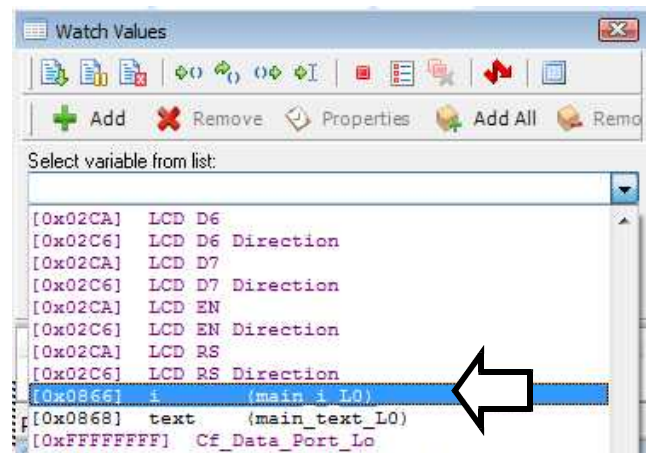
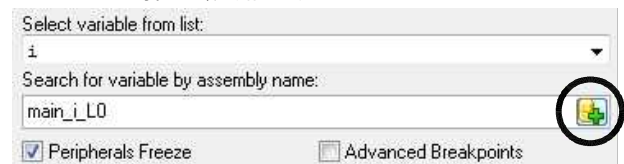
"for(i=0 ; i<13 ; i++){ " の行をクリックします。カーソルがこの行に移動していることを確認してください。

4 カーソル行までプログラムを一気に動かす方法でfor文位置までプログラムを移動させます。
メニューバーの"Run"→"Run to Cusor"をクリックします。キーボードのF4キーでも同様の操作ができます。
for文の一行が青色バーでハイライト表示されます。

5 LCDにはまだ文字は表示されていないはずです。
この状態で1行ずつ実行するステップ実行を行い、1文字ずつLCDに文字が表示されるのを確認しましょう。
キーボードのF8キーを押します。又はメニューバーの"Run"→"Step Over"をクリックします。F8キーを押すと、青色バーでハイライト表示された部分が移動します。
F8キーを2回押すと、LCDに"m"と表示されます。

6 同様にして、F8キーを何回も押すと、プログラムはFor文を繰り返し実行して、1文字ずつLCDに文字が表示されていきます。5文字程度表示されたら次の手順を実行します。
なお、F8キーを早く押してしまうと、ICD機能の処理が間に合わずエラーが表示されてしまうことがあります。F8キーを押してステップ実行を進める場合には、必ず青い実行行を示すバーが次の行に移動してからF8キーを押すようにしてください。

7 ICDを実行すると、Watch Valueウインドウが画面の右側に表示されます。ウォッチウインドウは、変数やレジスタの値をリアルタイムで閲覧できるウインドウです。ここでは変数iの値が1ずつ増えていく様子を確認しましょう。

8 変数iを追加します。
"Select Variable form list"のプルダウンをクリックして、その中から "[0xXXXX] i (main_i_L0)" と表示されている部分クリックします。※XXXXは値が入ります。

9 Addボタンを押して、追加します。

10 リストに追加されるとNameとValue、Addressが表示されます。
Valueの値は現在の変数iに格納されている数値です。
F8キーを押してFor文を実行すると、1ずつiの値が増えていく様子を見ることができます。

値が更新されると数値は赤文字で表示されます。

■ICD機能を使用しない場合の注意

ICD機能を使用する場合、コンパイル時のBuild typeをICD用に設定しました。このICD用に設定されて生成されたHEXファイルは、通常動作はしません。ICD機能を使用しない通常のHEXファイルをビルドしたい場合には必ずビルドする前に"Project Settings"の"Build type"から"Release"にチェックを入れて、ビルドを行ってください。

チュートリアル⑥　～UART通信とSPI通信の応用～

SPI通信は、デバイス間通信で使われることが一般的で、回路内であるICとIC(又はセンサ等)がデータを通信する時によく使用されます。同期式シリアル通信で、クロック信号に同期させて通信を行います。SPI通信の特徴は、送信と受信がクロック信号に同期して同時に行われることです。プログラムを書く場合には、仕組みをしっかりと理解しなくても書けてしまいますが、より理解を含めるためにはオシロスコープやロジックアナライザで波形を観察すると、動作内容が理解できます。

本dsPICF-400にはSPI通信を体験できるようにMCP23S17というポート拡張用のICが搭載されています。本チュートリアルではこのICをSPI通信で駆動させて、SPI通信の実験をしてみましょう。

■プログラムの内容

非同期式シリアル通信(UART)を使用して、RS232C通信でパソコンから8ビットのデータ(0～255)を送信すると、その値をSPI通信でポート拡張IC、MCP23S17に送信して、接続されているPORT0のLEDにその結果を出力するプログラムを作ります。

RS232C通信のプロトコルは通信速度9600bps,データ長8ビット長、パリティ無し、1ストップビットとします。なお、dsPIC30F4013には2系統のUARTを搭載していますが、UART1のピンはSPI通信のピンと競合しているため、このプログラムではUARTは2系統目のUART2を使用します。UART2のRXはRF4、TXはRF5にアサインされています。

■プログラムの作成

```
sbit  SPExpanderRST at RB11_bit;
sbit  SPExpanderCS  at RB10_bit;
sbit  SPExpanderRST_Direction at TRISB11_bit;
sbit  SPExpanderCS_Direction  at TRISB10_bit;

void init(){
  ADPCFG = 0xFFFF;
  SPI1_Init();
  Expander_Init(0);
  Expander_Set_DirectionPortA(0,0x0000);
  Expander_Set_DirectionPortB(0,0x0000);
  UART2_init(9600);
}

void main() {
  unsigned char rec_data;

  init();

  while(1) {
    if (UART2_Data_Ready()) {
       rec_data = UART2_Read();
       Expander_Write_PortA(0,rec_data);
    }
  }
}
```

■Pボードの設定

このプログラムを正しく動作させるため、下記のように設定します。

・SW7はRF4(3番)と、RF5(7番)のスイッチをONに設定。
・SW11はすべてのスイッチをON位置に設定。
・その他の全スイッチはOFF位置に設定。

■パソコン側の準備

このプログラムではRS232C通信を行います。パソコンのRS232Cポートと、dsPICF-400の"RS-232"とかかれたポートを全結線ストレートケーブルで接続してください。

■UARTで通信を行い動作を確認しましょう

UARTターミナルを使用してデータを送信して、正しくその値がLEDに出力されることを確認しましょう。

mikroCに付属のUARTターミナルを起動します。COMポートの設定及び通信速度9600bpsの設定を行い、"Connect"ボタンをクリックしてポートを開きます。

今回は、0～255の値を送信します(文字列の送信ではありませんので)ので、下記のように行います。

UARTターミナルの"Send"ボタンの下にある数値ボックスに数値を10進数で入力します。ここでは実験として170と入力します。

値を入力したら"Send ASCII"ボタンをクリックします。
クリックすると値がそのまま送信されます。
dsPICF-400基板上のP0.0～P0.7のLEDが点灯します。170は、2進数で表すと10101010ですので、LEDが交互に点灯します。0をUARTで送信すると全LEDが消灯します。

■プログラムの解説

sbit　SPExpanderRST　at　RB11_bit;
sbit　SPExpanderCS　at　RB10_bit;
sbit　SPExpanderRST_Direction　at　TRISB11_bit;
sbit　SPExpanderCS_Direction　at　TRISB10_bit;

SPI通信式のExpander IC用の組込関数のピンアサインを設定しています。MISO(Master In Slave Out)と、MOSI(Master Out Slave In)は、ハードウエアUSARTのピンにアサインされていますので、ピンの指定はできません。ここでは、チップセレクトピンのCSピンと、RSTピンのピンだけを指定しています。

SPI1_Init();

MPUのSPI通信ユニットの初期化をしています。

Expander_Init(0);

Expander関数の初期化をしています。引数の0は、デバイスのアドレスです。デバイスのアドレスは、ICのA0～A2ピンの状態を指定します。dsPICF-400では、J8のジャンパーで設定できます。J8の3つのジャンパーソケットが0に装着されていれば、MCP23S17のA0～A2はすべてGNDに接続されていますので、ここでは0を指定します。
アドレスを変えることで、複数のデバイスを同一バス上に取り付けることができます。

Expander_Set_DirectionPortA(0,0x0000);
Expander_Set_DirectionPortB(0,0x0000);

MCP23S17のPORTA及びPORTBの全ピンを出力に設定しています。引数の0は、デバイスのアドレス、0x00は出力を示します。

Uart2_Init(9600);

UART2の初期化を行う関数です。()内には通信速度をbps単位で記述します。ここでは9600bpsに設定しています。
dsPIC30F4013には2系統のUARTが搭載されています。

Expander_Write_PortA(0,rec_data);

SPI接続された指定アドレスのMCP23S17のPORTAに対して、指定したデータを出力する関数です。アドレスは0で、出力データには、UART通信で受信したデータの変数、rec_data変数を指定しています。そのため、受信したデータがLEDに表示されます。